Do jeito certo

Gestão de produtos no mundo das startups

Casa do Código

RIAN VAN DER MERWE

Sumário

- Parte 1 — Introdução
  - 1 Papéis e responsabilidades do gerente de produtos
- Parte 2 — Planejamento
  - 2 Descobrindo necessidades
  - 3 Descoberta do produto
  - 4 Roadmaps do produto
- Parte 3 — Execução
  - 5 Definindo um produto
  - 6 Design centrado no usuário e workflows
  - 7 Especificações
  - 8 Construção e lançamento
  - 9 Avalie e itere
  - 10 Gestão de produtos em metodologias ágeis
  - 11 Começando

Parte 1 — Introdução

No dia 9 de fevereiro de 2010, o vice-presidente de gestão de produtos da Google, Bradley Horowitz, subiu ao palco em uma conferência de imprensa para apresentar um produto chamado Google Buzz. Aclamado como "a nova forma de começar conversas sobre coisas que você acha interessante" (*Introducing Google Buzz*, http://smashed.by/buzz), Buzz foi uma rede social que vivia dentro do Gmail (*Google Gmail press conference*, http://smashed.by/buzz-release) e prometia características de uma "rica e rápida experiência de compartilhamento", e um filtro de relevância para destacar apenas os itens mais importantes.

Infelizmente, o Google não teve muito tempo para desfrutar da glória do lançamento da nova rede social. Quase imediatamente, a *feature* que deveria economizar o tempo dos usuários se tornou um pesadelo para a empresa. Uma vez que um usuário optava por entrar no Google Buzz, ele imediatamente passava a seguir, automaticamente, todo mundo com quem ele trocava e-mails ou mensagens com frequência.

O grande problema era que essa informação também se tornava disponível no perfil público do usuário do Google, o que significava que qualquer pessoa podia ver com quem aquele usuário interagia com mais frequência. Por fazerem a promessa de algo que "não requeria configuração", o Google inadvertidamente entrou em uma controvérsia sobre privacidade, da qual o produto nunca se recuperou completamente.

No dia 10 de fevereiro, um dia após, manchetes como "Google Buzz: um pesadelo de privacidade" (*Google Buzz: Privacy nightmare*, http://smashed.by/nightmare) e "CUIDADO: Google Buzz tem uma enorme falha com privacidade" (*WARNING: Google Buzz has a huge privacy flaw*, http://smashed.by/privacy-flaw) começarem a aparecer em blogs populares de tecnologia. Os artigos destinavam-se a prevenir os usuários quanto aos aspectos do Buzz que a interface havia falhado em explicar de maneira satisfatória: como a informação de um usuário era exibida publicamente, como

tornar a informação privada, e como editar a lista de pessoas que eram seguidas.

A equipe do Google Buzz reagiu imediatamente e trabalhou contra o tempo para resolver os problemas. Em 11 de fevereiro, o Google lançou uma feature para tornar mais fácil aos usuários a forma como não exibir a lista de pessoas que seguiam em seus perfis públicos do Google. Mas ele ainda seguia automaticamente todo mundo com quem o usuário trocava e-mails e conversava frequentemente. Logo, a crítica da imprensa continuou.

No dia 12 de fevereiro, a *Business Insider* publicou uma foto do Eminem mascarado empunhando uma serra elétrica sob a manchete "Blogueira indignada está sendo automaticamente seguida por seu ex-marido abusivo no Google Buzz" (*Outraged blogger is automatically being followed by her abusive ex-husband on Google Buzz*, http://smashed.by/stalking).

As coisas não estavam indo bem para a equipe do Google Buzz. Mas a única saída — por isso eles assim prosseguiram — era preparar uma sala de guerra dedicada a isso e colocar o Google Buzz em "Código vermelho" (*How Google went into 'Code Red' and saved Google Buzz*, http://smashed.by/code-red) para que as atualizações fossem liberadas o mais rápido possível.

No dia 14 de fevereiro, a equipe anunciou um monte de mudanças no Buzz (*A new Buzz start-up experience based on your feedback*, http://smashed.by/buzz-feedback), incluindo a maior delas: o Buzz não seguiria mais automaticamente usuários na configuração. Em vez disso, o Buzz mudaria para um modelo de sugestões automáticas, o que significava que os usuários teriam de explicitamente selecionar as pessoas que desejavam seguir.

Entretanto, para alguns, isso era muito pouco, e tarde demais. Em 16 de fevereiro, uma reclamação sobre os assuntos de privacidade foi solicitado contra o Google para a Federal Trade Comission (o pedido ao FTC foi estabelecido no dia 30 de março de 2011. Leia o

comunicado de imprensa aqui: http://smashed.by/ftc). Um estudante de direito da Harvard Law School solicitou uma ação coletiva judicial contra a empresa no mesmo dia.

Outraged Blogger Is Automatically Being Followed By Her Abusive Ex-Husband On Google Buzz

Nick Saint | Feb. 12, 2010, 11:31 AM | 10,502 | 57

Recommend 31 Share 1 Tweet 4 +1 5 Email More

The glaring privacy flaws in Google Buzz, which the company has failed to address fully, have hit home for one blogger who says she is now being auto-followed by her abusive ex-husband and his friends.

Harriet Jacobs (a name she goes by) writes on her blog that the people she receives email from most often include her ex-husband, his friends, and abusive commenters.

Figura 2: A imprensa de tecnologia teve algumas grandes manchetes por conta da história do Google Buzz (Fonte: Business Insider — http://smashed.by/stalking)

O caos claramente estabilizou e passou a diminuir. Eventualmente, a imprensa de tecnologia voltou seus olhos para o próximo caso cabuloso, e o Google Buzz continuou firme por um tempo. É claro, a história do Google Buzz não teve um final feliz. Em 15 de dezembro de 2011, ambos o Google Buzz e o Buzz API suspenderam suas atividades para todos os usuários, de modo que, no lugar deles, a empresa pudesse focar em sua rede social Google +.

O que o Google Buzz tem a ver com este livro? Bem, este é um livro sobre gestão de produtos e como desenvolver produtos de sucesso. E se você acabou de ler esta história e disse para si mesmo "Ei, deve ser divertido participar de uma experiência dessas!", então é quase certeza que gestão de produtos é a carreira para você.

Se você sentiu vontade de se esconder debaixo da mesa e tomar um trago, provavelmente é melhor considerar trilhar uma carreira diferente. Porque, por trás de toda decisão, todo erro, toda solução e todo lançamento de código no Google Buzz, havia um gerente de produtos que assumia a responsabilidade por cada consequência. Durante a maratona para o lançamento, os primeiros dias caóticos, bem como a batalha subsequente para convencer as pessoas a usarem o produto, o gerente de produtos permaneceu na lacuna entre o produto e o mercado para fornecer uma visão de longo prazo, assim como uma direção tática para manter as coisas andando para a frente.

Gestão de produtos é uma das carreiras mais exaustivas, emocionantes, estressantes e recompensadoras que existem. Não é para os fracos do coração. É para pessoas que desejam mover montanhas. Alguns são completamente engolidos por esse trabalho, mas outros encontram uma revigoração infinita e uma paixão pelo ritmo, pelo impacto, pela glória, e pelo enorme potencial tanto de fracasso como de sucesso que ele traz. Não há nenhum emprego como este, e este é um livro que vai ajudá-lo a fazer disso o seu trabalho.

Receio não ter conseguido vender o papel da gestão de produtos muito bem, por ter pulado direto nessa história do Google Buzz. Mas não há nada como uma boa dose de realidade para nos despertar para o que é ser um gerente de produtos: ter uma imensa tenacidade para desenvolver os melhores produtos no mundo. Mas, para equilibrar as coisas, vamos levar em conta a história de outro app, *Clear*, da empresa Realmac Software (http://smashed.by/clear).

Eu só posso imaginar as toneladas de complexidade caótica com que os gerentes de produtos, designers e desenvolvedores tiveram de lidar para chegar à simplicidade do Clear — um app de lista de afazeres para iPhone e Mac. A primeira vez que usei o app, a definição que Rebekah Cox deu para "design" veio à minha cabeça (*Web 2.0 Expo Presentation*, http://smashed.by/web-expo):

> *Design é um conjunto de decisões sobre um produto. Não é uma interface ou uma estética, não é uma marca ou uma cor. Design são as verdadeiras decisões.*

E a Realmac tomou algumas decisões difíceis que resultaram em um ótimo produto. Eu consigo imaginar as reuniões infinitas e discussões difíceis que acabaram por decidir quais features seriam incluídas na app.

Devíamos ter projetos e contextos? Não. E quanto a prazos finais e filtros? Não. Bem, por que não?! Porque Clear é uma lista de tarefas priorizadas que é rápida e fácil de editar. É isso. Nada a menos, nada a mais.

Para entender a beleza do Clear, além de olhar o que ele é, é importante olhar também o que ele não é. Clear é uma ótima maneira de ver e priorizar uma simples lista de tarefas, mas não é um substituto para complexos sistemas de organização de tarefas, como o Omnifocus. Mas usar o Omnifocus para tarefas simples, como agendar a revisão do carro ou encomendar café na loja, seria um exagero. Este é a lacuna que o Clear preenche.

O Clear foca em duas coisas que o torna superior aos apps similares:

- **Velocidade:** é realmente rápido. Assim que ele se inicia, você pode instantaneamente começar a digitar. Isso é crucial para poder captar rapidamente aquela coisa tão importante que você não quer esquecer.

- **Facilidade de edição:** é completamente baseado em gestos, sem frescuras, sem um design visual extravagante. Você toca, digita, desliza e fecha. Esses gestos são fáceis de se aprender e são intuitivos.

É realmente difícil resistir à tentação de construir um aplicativo que tenta resolver todos os problemas de todas as pessoas do mundo. O que torna a gestão de produtos do Clear tão impressionante é que eles atravessaram o inferno de dizer "não" para features potencialmente ótimas, e emergiram do outro lado, provavelmente chamuscados e surrados, mas também com um ótimo app para listagem de tarefas e rápida edição.

Você espera mais do seu app de listas de afazeres? Para isso serve o Omnifocus. E eles estão ok com isso. É algo para se ter orgulho.

Espero que a justaposição das histórias do Google Buzz e do Clear tenha mostrado que, embora gestão de produtos não seja sempre um trabalho fácil, ela também nunca é entediante.

Quando comecei em gestão de produtos, li um monte de livros sobre isso, mas nenhum deles me preparou para as realidades que eu enfrentaria uma vez que estivesse mergulhado nisso. O grande problema é que essa função é chamada de diferentes nomes — e se isso não é motivo suficiente para ficar confuso, algumas empresas definem gerente de produtos completamente diferente do que se entende em outros lugares.

Logo, você tem gerentes de programa, gerentes de projeto e analistas de negócio às vezes preenchendo a função de gerente de produtos. E aí você tem gerentes de produto em funções que seriam mais adequadamente chamadas de gerentes de produção ou *Product Owners*. Sei que às vezes nós ficamos presos em nossa missão de **definir a maldita coisa**, mas no caso de gestão de produtos, é um esforço válido, pois representa bem a selva lá fora.

Com esse plano de fundo, eu me programei para escrever este livro para alcançar três objetivos:

1. Definir os papéis e responsabilidades dos gerentes de produtos no contexto do desenvolvimento de software. Há tantas pessoas construindo produtos digitais e fazendo um monte de coisas que poderiam ser definidas como gestão de produtos, mas o que falta é uma definição holística e uma abordagem mais sistemática para a função. Eu espero preencher esta lacuna com este livro.

2. Explicar por que gestão de produtos é um papel essencial em qualquer organização, e quais características os gerentes devem procurar quando forem contratar gerentes de produtos.

3. Fornecer um framework e um guia prático para gestão de produtos estratégica; um framework que detalhe os elementos do planejamento do produto e a execução do produto que compõem o dia a dia de trabalho de um gerente de produtos.

É assim que o livro surgiu e como ele está estruturado. Na parte 1, darei uma visão geral do papel do gerente de produtos. O que é, para quem é, por que é importante e como isso se encaixa em uma empresa.

Na parte 2, discutirei elementos de planejamento de produtos. Como decidir o que desenvolver, como priorizar as necessidades de diferentes clientes e *stakeholders*, e como transformar isso em um plano estratégico e em uma trajetória (*roadmap*) para a entrega.

Na parte 3, falarei sobre como fazer a estratégia ser realmente aplicada durante a execução do produto. É aqui que nós entraremos no âmago da questão da definição do produto, testes das hipóteses, design e ciclos de entrega.

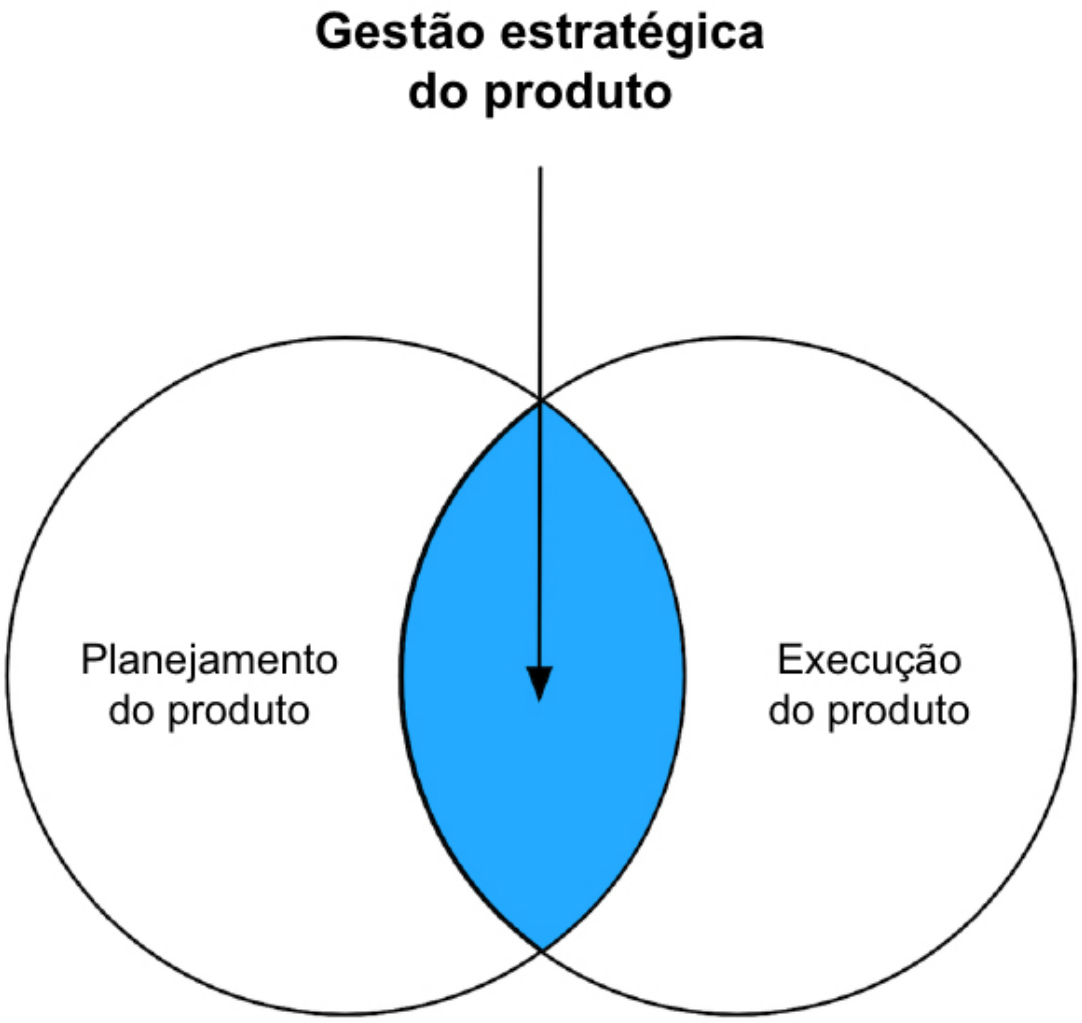


Figura 3: Gestão de produtos estratégica

Este livro é para qualquer pessoa que esteja pensando em fazer uma carreira em gestão de produtos, ou para aqueles que têm estado na área por questões estratégicas e estão procurando por um framework mais formal para o trabalho que eles realizam.

A maioria de nós vem para a gestão de produtos a partir de diferentes circunstâncias: backgrounds de design, análise de negócio, e desenvolvimento são os mais comuns. Portanto, este livro também é para quem quer continuar a trabalhar em suas especializações escolhidas, mas que gostariam de adquirir um

melhor entendimento de como seus trabalhos encaixam no grande quadro da estratégia de produto e entrega.

Em um âmbito pessoal, este livro é sobre muito mais do que estratégias, processos e metodologias. Nós vivemos em uma época incrível, dominada não pelos consumidores, mas por aqueles que criam software. Devido à internet e aos avanços na tecnologia digital, temos vasto acesso a ferramentas e *expertise* necessárias para criar novos produtos e melhorar aqueles que já existem. E eu quero ser parte desse movimento. Quero fazer o que eu puder para construir, e ajudar a nutrir essa paixão por criar em outras pessoas.

Meu medo — e o motivo pelo qual acredito que as ideias deste livro são tão importantes — é que nos faltem a paciência e os frameworks necessários para termos certeza de que entendemos os usuários, antes de começarmos a desenvolver produtos para eles. Faltam-nos as ferramentas apropriadas para poder devidamente planejar e executar nossas ideias de produto. Sobretudo, ainda não temos uma ampla adoção de times de pessoas que se dedicam excessivamente a essas coisas em nossas empresas: o produto, seus usuários, e como tornar ambos bem-sucedidos.

Meu desejo é que este livro incentive gerentes de produtos a desenvolver produtos melhores. Vou fornecer um framework estruturado — com a quantidade certa de processos — para deixar os gerentes de produtos confiantes de que estão construindo os produtos certos, no momento certo, para as pessoas certas.

Espero que qualquer um que trabalhe fazendo software, websites ou aplicativos mobile ache este livro útil para desenvolver produtos que sejam significativos para usuários, e que sejam sustentáveis como negócio.

Vamos começar!

Capítulo 2

Papéis e responsabilidades do gerente de produtos

O que são gerentes de produtos e o que eles fazem todos os dias? Boa pergunta.

O primeiro equívoco que temos de esclarecer é o que queremos dizer com a palavra *produto*. No contexto de desenvolvimento de software, o produto é o website, o aplicativo ou o serviço online com que usuários interagem. Dependendo do tamanho da empresa e de seus produtos, um gerente de produtos pode ser responsável por um sistema inteiro (como um aplicativo mobile) ou parte de um sistema (como o fluxo de checkout em um site de e-commerce, em todos os dispositivos).

Isso é confuso, porque na maior parte dos contextos um produto é algo que se vende para as pessoas. Particularmente no contexto do e-commerce, gestão de produtos frequentemente se confunde com gestão de categorias: o time que lida com recursos e propaganda dos produtos vendidos em um site de e-commerce. Então, sim, produto provavelmente não é a melhor palavra para isso. Mas é o que temos, e é a definição que usaremos para explorar este papel.

Para alcançar uma definição do papel da gestão de produtos, vamos começar observando a visão que Marc Andreesen deu para a única coisa que importa em um ambiente de startup (*Product/market fit*, http://smashed.by/market-fit; o negrito é meu):

> *A qualidade de um produto de uma startup pode ser definida por quão impressionante o produto é para um consumidor ou um usuário que realmente o use: o produto é fácil de usar? É rico em features? É rápido? Quão extensível ele é? Quão* ***POLIDO****? Quantos (ou antes, quão poucos) bugs ele tem?*
>
> *O tamanho de um mercado de startup são os números e a taxa de crescimento daqueles consumidores ou usuários daquele produto.*
>
> *[...]*
>
> *A única coisa que importa é fazer o produto e mercado se encaixarem.* ***O product/market fit significa estar em um bom mercado com um produto que o satisfaz.***

Embora Marc tenha escrito isso especificamente para o contexto de startups, a importância do product/market fit é uma verdade universal em qualquer empresa — seja se ela está colocando um novo produto no mercado, ou reformulando uma experiência já existente, ou qualquer coisa no meio disso. É um caminho universal para o sucesso, e o grande cerne pelo qual gerentes de produtos são responsáveis.

Tendo isso como plano de fundo, minha definição do papel do gerente de produtos é o seguinte:

> *A missão do gerente de produtos é alcançar sucesso no negócio, atingindo as necessidades dos usuários por meio de contínuo planejamento e execução de soluções de produtos digitais.*

Essa definição resume todas as coisas sobre as quais gerentes de produtos têm de se debruçar: o mercado-alvo; as complexidades do produto; o que o negócio precisa para atingir o sucesso; e como

medir esse sucesso. Além disso, isso engloba as três coisas que os gerentes de produtos nunca devem perder de vista:

- A medida definitiva de sucesso é o estado de saúde do negócio e, consequentemente, o valor que o produto fornece aos usuários.
- Tudo começa com um bom entendimento do mercado-alvo e suas necessidades, de modo que o foco se mantenha na qualidade da experiência do produto.
- Requere-se um contínuo ciclo de planejamento e execução para atingir essas necessidades do mercado de uma forma sustentável.

Como isso tudo traduz o que um gerente de produtos faz todos os dias? É sobre isso que se trata este livro. Em termos introdutórios, Marty Cagan tem uma ótima lista de tarefas comuns que pelas quais gerentes de produtos são responsáveis em seu livro *Behind every great product* (http://smashed.by/product-manager). A lista inclui:

- Identificar e avaliar a validade e a viabilidade das oportunidades de produtos.
- Assegurar-se de que o produto certo é entregue no momento certo.
- Desenvolver a estratégia do produto e o roadmap para o desenvolvimento.
- Guiar a equipe na execução do roadmap do produto.
- Colocar o produto em um pedestal internamente para o time executivo e colegas.
- Representar os clientes durante o processo de desenvolvimento do produto.

Vamos passar vários capítulos discutindo cada uma dessas atividades — e mais. Mas antes disso, precisamos responder algumas perguntas importantes. Primeiro, as empresas realmente precisam de gerentes de produto? E se concordamos com isso, quais as características de um bom gerente de produtos? E como esse papel se encaixa na estrutura organizacional? Vamos explorar essas questões.

2.1 Por que empresas precisam de gerentes de produtos

Gestão de produtos pode ser uma escolha difícil para algumas empresas. Objeções comuns incluem:

- "Temos diferentes pessoas na empresa que juntas preenchem todas essas funções como parte de seus papéis.
- "Eu não vejo como este papel vai nos trazer mais dinheiro."
- "Gerentes de produtos só vão nos atrasar."
- "Não quero ceder o controle do produto para outra pessoa." (Ok, esta aqui é mais comum ser apenas pensada do que dita em voz alta)

Estas parecem ser preocupações válidas a princípio, mas apenas se o papel não está bem entendido — ou se há gerentes de produtos ruins na empresa que perpetuam essas percepções.

A verdade é que, para ser efetivo, o papel da gestão de produtos para um produto ou área específica não pode ser preenchido por múltiplas pessoas. É essencial que o gerente de produtos veja o cenário todo — a visão estratégica, bem como os detalhes da implementação — para ajudá-los a tomar boas decisões para o produto. Se o conhecimento de diferentes partes do processo está

em cabeças de pessoas diferentes, a visão holística se esvai, levando todo o valor do papel.

Vejamos os dois maiores benefícios da gestão de produtos.

Gerentes de produtos garantem uma abordagem voltada ao mercado

O argumento chave a favor de gestão de produto é que ela ajuda empresas a serem guiadas pelas necessidades e objetivos do seu mercado-alvo, e não pelas forças da tecnologia ou por ondas de novidades. Como Barbara Nelson aponta em *Quem precisa de gestão de produtos?* (*Who needs Product Management?*, http://smashed.by/pmanagement):

> *É muito mais fácil identificar problemas do mercado e resolvê-los com tecnologia do que encontrar compradores para a tecnologia existente.*

Se feito da maneira correta, atuar voltado ao mercado resulta, no longo prazo, em um negócio sustentável e lucrativo, já que a empresa mantém seu foco em resolver problemas do mercado, em vez de procurar coisas a ver com as mais novas tecnologias.

Guiar-se pelo mercado é importante pois empresas assim provaram ser mais lucrativas do que aquelas dirigidas por outros fatores (31% mais lucrativo, de acordo com a pesquisa *Managerial representations of competitive advantage*, de George S. Day e Prakash Nedungadi).

Isso não significa que você vai se focar exclusivamente em mudanças incrementais em vez de inovação de produto. Identificar problemas do mercado não se trata apenas de encontrar questões a melhorar (por exemplo, "60% de usuários saem desta página, vamos consertar isso"), mas também de criar produtos novos que satisfaçam necessidades não atendidas ("Celulares não prestam,

vamos criar um novo"). Uma das coisas que discutiremos mais à frente é como realizar pesquisas que descubram a necessidade por trás das features, para ajudar empresas tanto em inovação como em iterações.

Gerentes de produtos melhoram o time-to-everything

O segundo maior benefício da gestão de produtos é que ela reduz o tempo necessário para alcançar os objetivos da empresa. Um processo de desenvolvimento de produtos apropriado e bem definido, dirigido por gerentes de produtos efetivos, melhora tanto o *time-to-market* como o *time-to-revenue*, ou seja, quanto tempo demora para alcançar o mercado e para obter retorno financeiro, respectivamente.

O motivo pelo tempo de resposta mais rápido é que gerentes de produtos são responsáveis por descobrir o que vale ou não a pena desenvolver. Isso implica que menos tempo é gasto na abordagem *spaghetti* para desenvolvimento de produtos (jogando as coisas na parede e vendo o que gruda), e mais tempo destina-se a construir produtos validados no mercado.

Essa abordagem também provê foco para uma empresa, de modo que mais pessoas podem se dedicar a produtos com probabilidade de sucesso do que dividi-las igualmente em projetos que ninguém tem certeza de que se encaixará na relação entre produto e mercado.

2.2 Características de um bom gerente de produtos

Agora que já falamos da importância da gestão de produtos, a próxima questão é: *"Quem são essas pessoas?".*

A maioria de nós é familiarizada com a ideia de pessoas em formato T: aqueles com profundo conhecimento em uma ou duas áreas, combinado com um entendimento razoável de uma variedade de disciplinas relacionadas ao campo de foco principal. Em 2009, Bill Buxton escreveu um artigo interessante para a Businessweek, no qual ele clama por mais pessoas em formato I (*Innovation calls for I-shaped people*, http://smashed.by/ishape):

> *Essas pessoas têm seus pés firmes no solo do mundo prático, e ainda se esticam o suficiente para alcançar a cabeça nas nuvens quando precisam. Além disso, eles alcançam simultaneamente todo o espaço no meio do caminho.*

Essa é uma boa descrição da mistura única de habilidades que gerentes de produtos precisam ter. Primeiro, eles precisam ter a cabeça nas nuvens. Gerentes de produtos precisam ser líderes que conseguem olhar para o futuro e pensar estrategicamente. Precisam ser capazes de desenvolver a visão de onde cabe um produto — e de comunicar esta visão efetivamente.

Além disso, eles precisam mostrar a seus times como planeja chegar a essa visão. E realmente quero dizer "mostrar", através de *sketches*, protótipos, *storyboards* , ou qualquer coisa necessária para transmitir a mensagem. Bons gerentes também são flexíveis e mudam o curso quando necessário, talvez quando há uma grande mudança nas exigências ou expectativas do mercado, ou quando uma ótima oportunidade de negócio se apresenta.

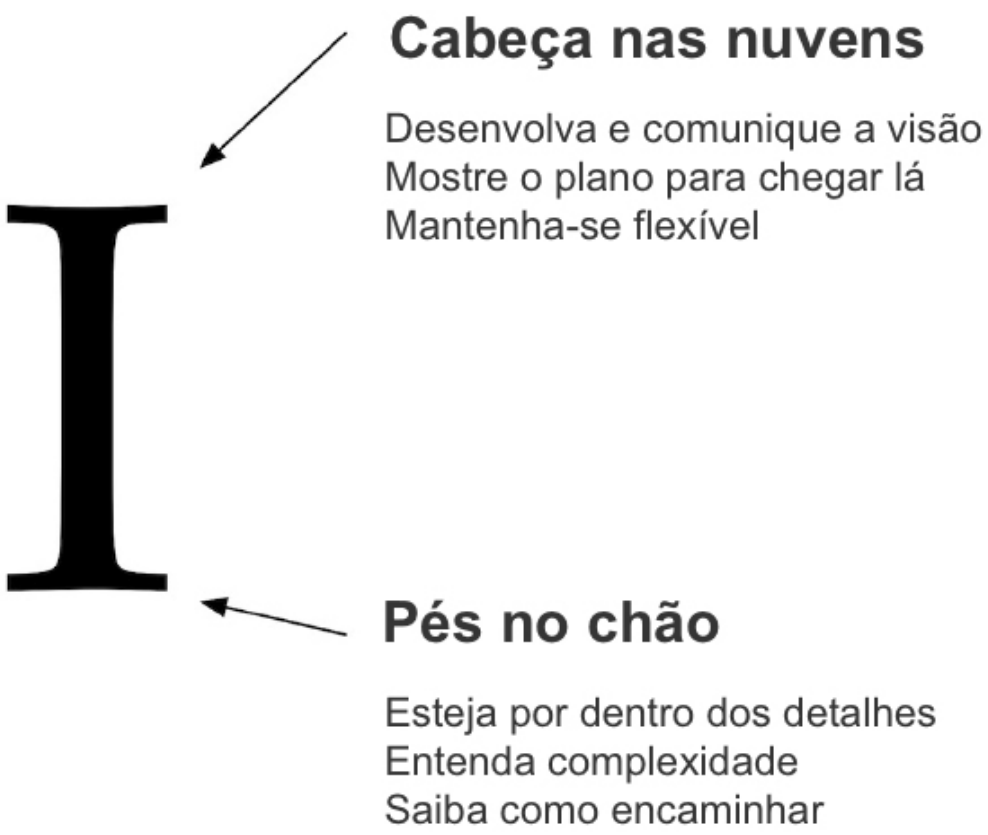


Figura 2.1: Pessoas no formato I (I-shaped people)

Por outro lado, bons gerentes de produtos também têm os pés no chão. Eles prestam atenção aos detalhes e conhecem seus produtos de trás para a frente. São os maiores usuários do produto — e os maiores fãs e críticos. Eles entendem cada aspecto da complexidade que precisa ser trabalhada em cada decisão de produto. E são capazes de tomar essas decisões rapidamente com base em todas as informações que têm ao seu dispor.

Principalmente, gerentes de produtos sabem entregar. Sabem como executar e recrutar um time para conduzir produtos e melhorias no mundo agora, onde o mercado-alvo pode usá-los e fornecer feedback.

Em resumo, eles são tanto visionários como fazedores. Gerentes e criadores. E eles precisam se mover sem problemas entre os dois extremos, às vezes em um piscar de olhos. Parece difícil? Isso é apenas o começo. Vejamos outras características de bons gerentes de produtos:

- Líder e colaborador
- Comunicador e negociador
- Apaixonado e empático
- Qualificado e curioso
- Confiável e ético
- Responsável e flexível

Líder e colaborador

Ser um líder e um colaborador ao mesmo tempo pode ser um equilíbrio difícil de atingir. O primeiro desafio é que a colaboração frequentemente é confundida com consenso. Mas este não é o caso.

Culturas de consenso geralmente produzem produtos fracos e desinteressantes. Produtos em que infinitas rodadas de discussões desgastaram a ideia inicial e a tornaram apenas uma sombra do que era antes. Culturas de consenso também desgastam os times que trabalham no produto, pois ninguém realmente pega aquilo que quer, apenas parte daquilo.

Colaboração é diferente. Em culturas de colaboração, pessoas entendem que mesmo que todos tenham uma voz, não é todo mundo que pode decidir. As pessoas podem dar suas opiniões, discutir passionalmente sobre como acreditam que as coisas deveriam ser feitas, e tentar negociar compromissos. Mas isso certamente não quer dizer que todos têm de concordar com cada decisão.

O primeiro passo para construir uma cultura de colaboração é ser um bom líder. Como você provavelmente já suspeitava, o gerente de

produtos é o tomador de decisões finais. Mas isso só funciona se ele é um líder confiável e respeitado na organização — alguém que consiga estimular o time sobre uma visão, bem como tomar as melhores decisões para o benefício da empresa e de seus clientes. Bons líderes também admitem prontamente quando tomaram uma decisão errada, assumem a responsabilidade e fazem o que for necessário para consertar o erro.

Este não é um livro sobre liderança — há vários deles por aí. Mesmo assim, ainda vou compartilhar alguns conselhos de liderança do escritor e aviador francês, Antoine de Saint Exupéry. Uma paráfrase do poema *Dessine moi un bateau*, encontrado em *Citadelle* (Cidadela, em português), de 1948:

> *Se você quer construir um navio, não convoque homens para juntar madeira, e não lhes atribua tarefas e trabalho. Antes, ensine-os a desejar a infinita imensidão do mar.*

O que significa "a infinita imensidão do mar" em seu contexto? Em vez de falar para as pessoas construírem um monte de features, como você pode inspirá-las a pensar em como o produto vai ajudar seus usuários a atingirem seus objetivos? É assim que você conseguirá unir equipes em volta de uma visão comum.

Então, como um bom líder nutre esse tipo de cultura de colaboração? Criando o ambiente certo e processos que permitem que a colaboração se retroalimente, e entendendo que cada pessoa é diferente e reagirá de maneira imprevisível em algum momento.

Para criar o ambiente certo e processos de colaboração, foque primeiro no ambiente físico. Assegure-se de que os espaços de trabalho permitem tanto discussões repentinas com os membros do time quanto a habilidade de expulsar todo mundo para trabalhar livre de distrações por um período de tempo.

O escritório do MailChimp é um ótimo exemplo. O time criou um ambiente de trabalho colaborativo (*New MailChimp: collaboration by design*, http://smashed.by/collaborationbydesign), baseado nos seguintes princípios:

- **Combinação e polinização cruzada —** Em vez de segregar equipes, misture pessoas com base em suas personalidades e nos projetos em que estão trabalhando. Isso leva a discussões valorosas que talvez não aconteceriam se cada um estivesse preso em seu próprio cubículo.
- **Facilitar movimento —** Escrivaninhas, sofás, mesas altas: todos eles são modos de encorajar pessoas a moverem-se e trabalharem juntos quando necessário.
- **Ideias em todos os lugares —** Cubra as paredes e quadros brancos com esboços, rascunhos, designs, listas de priorizações, roadmaps. Isso não apenas contribui para melhor comunicação, mas também deixa a porta aberta para quem quiser melhorar as ideias em que as pessoas estão trabalhando.
- **Criar convergência —** Um espaço comum para almoço (e café!) é importante, porque facilita o encontro de pessoas, mesmo que elas normalmente não trabalhem juntas em um projeto. Isso, mais uma vez, pode resultar em grandes ideias e perspectivas.
- **Crie retiros —** Um espaço de colaboração bombando tem ótima energia, mas às vezes pode ser uma distração. Ocasionalmente, indivíduos ou times precisam de um espaço mais quieto para trabalhar. Portanto, tenha certeza de ter salas de reunião ou retiros de silêncio onde não haverá nenhuma interrupção.

Lugares de trabalho são mais importantes do que pensamos. Não poupamos esforços ao criar um espaço receptivo e criativo no estúdio em que eu costumava trabalhar, e podemos ver seus

resultados. A maior parte de nossos clientes prefere vir até nós quando temos reuniões, e eles mencionam dois motivos: um café excelente (exageramos um pouco em relação ao café), e uma ótima atmosfera onde trabalhar.

Steve Jobs compreendia o valor dos locais físicos muito bem. Ele disse o seguinte sobre o design do novo campus da Pixar (Issaacson, Walter: *Steve Jobs*):

> Se um prédio não encoraja [a colaboração], você perderá muito da inovação e da mágica que se despertariam com a alegria. Então, projetamos esta construção para fazer as pessoas saírem de seus escritórios e misturarem-se no atrium central com outras pessoas que talvez eles nem vissem.

É claro, o espaço físico é apenas uma parte da equação. Muito do trabalho acontece remotamente agora, e temos ferramentas suficientes à nossa disposição para tornar isso uma experiência efetiva e recompensante para todos os envolvidos. Desde ferramentas de comunicação como o Campfire, o Hipchat e Slack, até ferramentas colaborativas de gestão de projetos, como Trello, Basecamp e Jira, até repositórios de código-fonte, como GitHub e Bitbucket. Não há mais desculpas para forçar todo mundo a estar no mesmo ambiente físico todo o tempo. Ainda há muito valor em conversar com outra pessoa pessoalmente, e colaborar em certas áreas do processo, mas até isso pode acontecer em espaços digitais.

Então, o que vem depois de trabalhar os ambientes físicos e digitais? Uma palavra temida... Muitas pessoas escutam "processo" e pensam que é um sinônimo de "coisas que preciso fazer em vez de trabalhar". Mas nós vamos falar bastante sobre apropriação, ou processos de alta fidelidade neste livro.

Citando Michael Loop: *"Engenheiros não odeiam processos. Ele odeiam processos que não se sustentam"* (*The process myth*,

http://smashed.by/process-myth). Quando se trata de criar uma cultura de colaboração, há vários processos que podem tornar a vida mais fácil para a equipe toda: processos sustentáveis.

Um processo de colaboração essencial para se acertar é ter sessões regulares de feedback sobre o design, o desenvolvimento e as decisões de negócio. O problema é que sessões de feedback podem sair de rumo rapidamente, porque não somos muito bons em fornecer (ou receber) feedback. Somos inclinados a ver os aspectos negativos da ideia de outra pessoa primeiro, então frequentemente pulamos direto para a destruição. Isso coloca a pessoa que vai receber o feedback em um modo defensivo logo de cara, o que comumente dá início a uma espiral negativa para argumentos inúteis e desconfianças.

Entretanto, existe um jeito melhor. Em uma entrevista sobre criticismo e julgamento, o filósofo francês Michel Foucault explicou o propósito de cada boa crítica. Para ele, o criticismo não devia focar no que não funciona, mas em como você pode transformar as ideias dos outros para torná-la melhor:

> Não posso me impedir de pensar em uma crítica que não procuraria julgar, mas procuraria fazer existir uma obra, um livro, uma frase, uma ideia; ela acenderia os fogos, olharia a grama crescer, escutaria o vento e tentaria apreender o voo da espuma para semeá-la. Ela multiplicaria não os julgamentos, mas os sinais de existência; ela os provocaria, os tiraria de seu sono. Às vezes, ela os inventaria? Tanto melhor, tanto melhor. A crítica por sentença me faz dormir. Eu adoraria uma crítica por lampejos imaginativos. Ela não seria soberana, nem vestida de vermelho. Ela traria a fulguração das tempestades possíveis. (Em *O filósofo mascarado*, entrevista com C. Delacampagne. fevereiro de 1980, Le monde. Nº 10.945, 6 de abril de 1980: Le monde-dimanche. ps. I e XVII.)

Mantendo este propósito em mente, eu particularmente gosto do processo usado por Jared Spool e sua equipe na UIE (*Moving from critical review to critique*, http://smashed.by/critical-review). O time utiliza-o especificamente para críticas de design, mas pode ser aplicado genericamente para qualquer tipo de sessão de feedback. Eis como o processo funciona:

1. A pessoa apresentando a ideia ou trabalho descreve o problema que está tentando resolver.

2. Se todos concordam com o problema, o time prossegue. Contudo, se não há acordo sobre o problema a ser solucionado, é necessária alguma discussão para esclarecer. Mas esperamos que este passo não seja necessário.

3. A seguir, o apresentador comunica sua ideia ou mostra seu trabalho para o time. O objetivo não é apenas mostrar o produto final, mas explicar o pensamento por trás da ideia ou da entrega. O apresentador deve permanecer focado em como a ideia vai resolver o problema com que todos concordaram.

4. O primeiro passo para o feedback é as pessoas na sala apontarem o que gostam naquela ideia. Esse não é um artifício para usar o método de sanduíche de porcaria (você sabe: começar e terminar com algo positivo, e jogar as tripas no meio). Em vez disso, este passo ajuda a ressaltar qual direção é desejável como solução para o problema.

5. O próximo passo são as críticas. Entretanto, não como ataques diretos ou frases como "Eu não gosto...", mas como questionamentos sobre a ideia. Os membros da equipe indagam se uma solução diferente foi considerada, e assim por diante. Isso dá ao apresentador a chance de responder se já pensaram nesse assunto, ou então, anotar para pensar na questão na próxima iteração.

6. Ao fim da reunião, o time revê as anotações, especialmente o que todos gostaram, e quais questões tiveram. O apresentador

então vai trabalhar na próxima iteração da ideia.

Como gerente de produtos, você deve se assegurar de que as sessões de feedback aconteçam, e que sejam respeitáveis e úteis. Scott Berkun possui um ótimo conjunto de regras que precisamos lembrar sobre críticas (*How to give and receive criticism*, http://smashed.by/berkun):

- **Assuma o controle do processo de feedback —** Feedback é algo que você deve fazer acontecer, pois assim ele acontece da sua maneira e de um jeito que melhore o produto. Se você apenas esperar que ele aconteça sozinho, isso só vai ser em reuniões quando você não está preparado; você ficará na defensiva e o foco vai mudar de produto para política.
- **Escolha seus parceiros —** Algumas pessoas são melhores em fornecer feedback do que outras. Encontre parceiros de feedback com uma experiência relevante que você precisa para tornar o produto melhor.
- **Empenhe-se em ouvir tudo, informalmente e antecipadamente —** Não espere até que o produto esteja quase pronto antes de obter feedback. Discuta ideias, conceitos e esboços bem antes de discutir tecniquês e código.

O objetivo da colaboração é que as ideias se tornem melhores, com base nos melhores pensamentos dos diferentes pontos de vista. Enquanto as pessoas confiarem que o tomador de decisão (você, caro gerente de produtos) tem os interesses do produto e da empresa como foco principal, eles não terão problemas em não ser sempre do jeito deles vez ou outra.

Seja confiante, confiável e decisivo — e assegure-se de que todo mundo se sente confortável em trazer suas opiniões para o time. O livro *Crucial conversations: tools for talking when stakes are high* (Kerry Patterson, Joseph Grenny, Ron McMillan e Al Switzler, 2011) é uma ótima fonte sobre como construir este tipo de ambiente.

Um último comentário sobre colaboração. A empresa de design e estratégia Cooper tem um ótimo conjunto de princípios-guias para garantir boa colaboração por dentro e através dos times (*Better together; the practice of successful creative collaboration*, http://smashed.by/better-together). Algumas dessas diretrizes — mais uma vez ajustadas para um contexto mais amplo do que apenas design — são:

- **Não trabalhe sozinho —** É preciso haver um fluxo e fluidez natural na maneira como o time trabalha. Parte do trabalho será feita por contra própria (documentação, criação no Photoshop, e por aí vai), mas este período de trabalhar sozinho deve sempre ser seguido por um tempo unido à equipe, para fornecer críticas e empurrar as ideias para a frente.

- **Exteriorize o pensamento —** É importante compartilhar as iterações iniciais de uma ideia com o time. Falaremos disso em detalhes depois, mas uma vez que o time começa a iterar sobre uma ideia, torna-se muito mais difícil mudar de rumo. Conversar sobre ideias em estágio inicial ajuda a equipe a considerar uma variedade de alternativas.

- **Presuma valor, mesmo quando não está óbvio —** Este é truque do "Sim, e..." que eles ensinam nas aulas de *brainstorming*. Em vez de procurar falhas em algo, tente encontrar formas de continuar construindo com base nas ideias compartilhadas.

- **Deixe os egos na entrada —** Este é frequentemente o componente da colaboração mais difícil, mas também o mais essencial. Não é apenas importante fornecer bom feedback, mas também é essencial recebê-lo bem. Ser bom em receber feedback significa ouvir atentamente, fazer anotações, pedir mais detalhes quando necessário e, sobretudo, ter humildade suficiente para reconhecer que você estará errado uma vez ou outra.

É muito melhor consertar erros mais cedo no processo de desenvolvimento do que teimar no seu ponto e a coisa falhar quando já estiver viva. Deixar os egos na entrada significa que todo mundo pode ser bom, já que você está menos propenso a cometer erros.

Tudo isso é muito mais fácil falar do que fazer, é claro. Gerentes de produtos devem dirigir o time através do processo de colaboração. Às vezes, a confiança não estará lá no começo. Tudo bem, confiança leva tempo. Viva estes valores, guiados pelo exemplo, que a cultura virá.

Comunicador e negociador

Talvez um nome mais adequado para esta seção seja "hipercomunicador e negociador", porque se há algo que um gerente de produtos nunca pode se cansar de fazer é contar para as pessoas o que está acontecendo. Mas em vez de enviar toneladas de e-mails, há um jeito melhor — sobre o qual discutiremos bastante —, que é trabalhar abertamente o máximo possível.

Assegure-se de que notas, esboços, planos e estratégias estejam bem acessíveis para todo mundo na empresa, em qualquer momento. Isso pode acontecer tanto em quadros brancos pelo escritório, ou em arquivos wiki e pastas de projetos. Trabalhar em público traz o benefício das conversas contextuais: todos os comentários e decisões estão em um só lugar em vez de espalhados em múltiplos e-mails (ou, pior ainda, em reuniões das quais ninguém se lembrou de anotar nada).

Ao ser um gerente de produtos, às vezes você pode sentir-se despedaçado. A maioria dos stakeholders apenas mantém interesse profundo em seu próprio departamento (como bem deveriam, e são pagos para lutar por aquilo com o que se importam). Por outro lado, o gerente de produtos deve negociar a melhor solução dentre todas as diferentes direções que stakeholders querem tomar, e então

comunicar esta decisão eficientemente sem alienar as pessoas que não entendem isso. Não é uma tarefa fácil.

Figura 2.2: O que gestão de produto às vezes parece (painel central do tríptico Martírio de Santo Hipólito, de Dieric Bouts, 1468)

A frase que a comunidade de design tem adotado para se referir ao difícil processo de gerir as expectativas (e asserções) de vários stakeholders é: design por comitê (*design by committee*). Mais uma vez, uma cultura mais genérica de decisões-por-comitê (*decisions-*

by-committee) é bastante difundido, principalmente em grandes empresas. Eu sempre gostei da abordagem que Speider Schneider propõe em seu artigo *Why design-by-committee should die* (http://smashed.by/design-committees):

> *A resposta sensata é ouvir, absorver, discutir, poder defender toda decisão de design com clareza e razão, saber quando se engajar em uma batalha e quando deixá-la.*

Isso não é tão fácil quanto parece, portanto, com o tempo desenvolvi as seguintes diretrizes para lidar com decisões por comitê de um jeito sistemático:

Responder a todo e cada feedback

Responder a cada questão, cada ideia, crítica e demanda leva tempo. Mas deixar sem resposta vai fazer gastar ainda mais tempo e energia pelo caminho. Uma coisa é quando alguém ouve sua ideia e não a usa. Outra coisa totalmente diferente é quando alguém nem ao menos a ouve. Em vez de lidar com as ramificações políticas de não ouvir as pessoas, é melhor tomar o tempo preciso para responder carinhosamente quando alguém dá um feedback (não importa quão inválido) ou acrescenta alguma ideia.

Perceba qual feedback está sendo incorporado

Quando você implementa uma boa ideia, não o faça em silêncio. É uma oportunidade de mostrar que você é flexível e aberto a bom feedback. Então faça as pessoas saberem quando e como suas ideias estão sendo usadas. Além disso — o que eu nem deveria ter de dizer —, não leve crédito pela ideia de outra pessoa.

Quando feedback não está sendo incorporado, explique o motivo

Praticamente falando, a maior parte dos feedbacks que você recebe não será incorporada ao produto. É importante não varrer essas

decisões para debaixo do tapete. Ao se forçar a ser claro e franco sobre feedback que não é implementado, você também se força a pensar em sua decisão e em defendê-la apropriadamente.

Às vezes, você até vai perceber que aquilo que você inicialmente descartou por ser uma ideia ruim, no fim das contas, seria um aprimoramento. O maior benefício de fazer isso é que as pessoas geralmente estão ok se seus feedbacks não estão sendo usados, desde que tenham sido ouvidos, e que haja uma boa razão para essa decisão.

Use um stack de validação para argumentar suas decisões

No livro *Undercover User Experience Design* (http://undercoverux.com/), Cennydd Bowles e James Box explicam o stack de validação de experiência de usuário, um método que pode, mais uma vez, ser aplicado genericamente para argumentar sua decisão de produto. Ao defender uma decisão, sempre tente usar evidências em forma de dados coletados diretamente dos usuários, tais como testes de usabilidade, ou web analytics.

Se você não tem dados diretos, procure por pesquisas terceirizadas — ou pesquisas prévias que você fez, ou pesquise em áreas similares, que podem ser aplicadas para os problemas que você está tentando resolver. Se nada disso funcionar, persuasão, psicologia, e por aí vai, podem ser muito úteis para explicar por que você tomou certas decisões.

Estas diretrizes devem tornar mais fácil a negociação entre diferentes necessidades e pedidos dos stakeholders. Mas lembre-se da recomendação de Speider, em seu artigo: *você deve saber quando engajar em suas batalhas e quando deixá-la. Esta é a arte de ser um bom negociador e comunicador.*

Apaixonado e empático

Gerentes de produtos possuem uma paixão e um profundo respeito por produtos bem feitos e com bom design — tanto físico como

digital. E eles vivem para criar produtos como esses. Eles são as pessoas que vão a festas e não conseguem ficar quietos sobre um novo aplicativo ou site que viram recentemente; ou mais provável ainda, ficam falando sobre o negócio em que estão trabalhando e como ele é legal.

Mas eles não são apenas apaixonados pelo produto: eles são apaixonados pelas pessoas que usam os produtos também. Eles têm um grande entendimento de seu mercado: os valores, prioridades, percepções e experiências dos seus clientes.

Paixão por um produto é inútil sem empatia — uma grande preocupação e compreensão — sobre as pessoas que usam o produto. Eu não acho que é possível construir ótimos produtos sem a habilidade de entrar na mente das pessoas que o usam. Se queremos antecipar o que elas vão fazer e guiá-las ao longo do caminho, empatia não é algo opcional.

Qualificado e curioso

Gerentes de produtos geralmente vêm de diferentes backgrounds, como experiência de usuário, programação ou análise de negócio. Para aplicar este conhecimento de suas áreas de modo mais geral — em outras palavras, tornar-se mais *tipo I*, eles precisam estar não apenas extremamente confortáveis em seu conjunto de habilidades, mas também devem conseguir aprender novas habilidades rapidamente (e sob grande pressão).

Esta combinação de ser qualificado tanto quanto capaz de continuar aprendendo indica que os gerentes de produtos devem ser insaciavelmente curiosos em tudo o que fazem. Por quê? Cap Watkins fala sobre isso muito bem (*Curiosity required*, http://smashed.by/curiosity):

"[...] se você é intensamente curioso, eu tendo a me preocupar menos com as outras habilidades. Mais e mais vezes vejo grandes designers adquirirem novas habilidades e ampliar os limites do que pode ser feito, por meio de pura curiosidade e força de vontade. Curiosidade nos força a ficar acordados a noite toda aprendendo uma nova técnica no Photoshop. Ela nos acorda no meio da noite, porque não conseguimos nos esquecer daquele problema de iteração que ainda não resolvemos. Eu honestamente acho que esse é o único e mais importante traço que um designer (ou qualquer um trabalhando com tecnologia) pode ter."

Bons gerentes de produtos fazem o que for necessário para tornar o produto bem sucedido. Eles constantemente se preocupam com os menores detalhes tanto quanto se preocupam com a maior das questões estratégicas. E em vez de estarem sobrecarregados pela quantidade de coisas a serem feitas, sua curiosidade os fazem permanecer comprometidos e tornarem-se tão qualificados quanto possível para tomar as decisões certas.

Confiável e ético

Bons gerentes de produtos constroem confiança com seus times por meio de cada decisão tomada. Para ser digno de confiança, eles devem ser justos (veremos mais sobre isso adiante), consistentes, e sempre assumir a responsabilidade por suas decisões. Eles também devem admitir quando estão errados, o que pode ser bem difícil, mesmo na melhor das hipóteses.

Em um extremo, gerentes de produtos precisam estar confiantes sobre as decisões tomadas. Eles devem continuar aprendendo e crescendo, aprimorando seu ofício constantemente. Teoria sólida e técnica excelente devem se tornar tão intrínsecas que simplesmente viram uma segunda natureza, alicerces de tudo o que fazem.

Mas, igualmente importante, eles devem estar abertos à possibilidade de que algumas de suas decisões estejam erradas. Aliás, eles devem receber isso bem. Devem seguir uma medida de dúvida própria cada vez que apresentam uma nova solução para o time e para o mundo.

Admitir que a ideia de outrem é melhor do que a sua própria e realizar alterações com base em uma boa crítica fazem maravilhas para melhorar o produto — e para erguer confiança entre o time. No contexto de design, John Lilly verbaliza isso de uma forma que deveria se tonar um mantra para todos os gerentes de produtos: *Design like you're right; listen like you're wrong* (http://smashed.by/you-right).

No que tange ao assunto da confiança, os melhores gerentes de produtos são aqueles guiados por um ponto de vista bastante forte e ético sobre o mundo. Discutir sobre ética pode me deixar em apuros, mas seria errado se eu nem ao menos tocasse no assunto.

O ponto é que não estamos apenas fazendo produtos aqui. Estamos deixando uma marca no mundo, e temos a oportunidade de fazer com que seja distintivamente boa. Deixar este lugar melhor do que o encontramos. Talvez ninguém o diga melhor do que Mike Monteiro, em *Design is a job* (http://smashed.by/design-job):

> *"Eu incito cada um de vocês a buscar projetos que tornem o mundo um lugar melhor daquele que você encontrou. Costumávamos fazer projetos para chegar à lua; agora projetamos planos para nunca ter de sair da cama. Você tem o poder de mudar isso."*

Como encontrar projetos e problemas que se encaixam neste critério? Uma das maneiras é tomar cuidado com o que Paul Graham chama de *schlep blindness*, ou "cegueira inercial" (http://www.paulgraham.com/schlep.html). Isto é, nossa inabilidade

de identificar problemas difíceis de resolver — principalmente porque não estamos conscientemente procurando por eles.

Como Paul recomenda que combatamos isto? Em vez de se perguntar qual problema você deve resolver, pergunte-se qual problema você gostaria de que outra pessoa resolvesse para você.

Outra ótima fonte de projetos e ideias dignos é o campo de empreendedorismo social (buscar soluções inovadoras para problemas sociais). Meagan Fallone tem uma boa visão geral da natureza e da importância deste tipo de trabalho (*Technology is useless if it doesn't address a human need*, http://smashed.by/useless-tech):

> *"Em troca, nós podemos ensinar o Vale do Silício sobre a relação humana entre a função de design e o impacto que isso tem na qualidade de vida de um ser humano. Não consideramos os usuários de tecnologia como "clientes", mas como seres humanos cujas vidas podem ser melhoradas pela desmistificação e acesso à tecnologia. Caso contrário, a tecnologia não teria lugar entre as necessidades básicas humanas que vemos no mundo em desenvolvimento.*
>
> *Projetos de tecnologia sustentável devem visar a mudanças reais; isso é indiscutível para nós. Iniciativas sociais, por conta própria, possuem a responsabilidade de garantir sustentabilidade e impactar em cada aspecto possível de nosso trabalho".*

O livro *Wicked problems* (https://www.wickedproblems.com/) é um ótimo ponto de partida para ideias sobre como dirigir nossos esforços visando um trabalho significativo.

É claro, todos nós temos definições diferentes de trabalho que seja importante e empurre a sociedade para a frente. Tudo bem — o que

importa é pensar nisso, e ter definições e limites claros para o trabalho com qual você quer se envolver.

Responsável e flexível

Tem um conhecido provérbio que gerentes de produtos usam para explicar seu papel, na tentativa de angariar alguma simpatia. O ditado diz que a parte mais difícil de ser um gerente de produtos é que você tem toda a responsabilidade, mas nada da autoridade.

O que isso quer dizer é que, mesmo que gerentes de produtos sejam responsáveis pelo sucesso ou falha do produto, eles geralmente não têm ninguém os informando. É por isso que comunicação e colaboração são características tão cruciais para gerentes de produtos eficazes.

O lado perigoso de ter toda a responsabilidade por um produto é que isso pode levar à rigidez: uma inabilidade de deixar algumas tarefas que poderiam ser facilmente delegadas, bem como uma teimosia em insistir em um plano original, mesmo quando as circunstâncias mudaram e um novo caminho é necessário.

É por isso que gerentes de produtos devem permanecer flexíveis. Planejamento é extremamente importante — passaremos um bom terço do livro falando sobre isso. Mas uma parte essencial do planejamento é permitir que a informação certa possa mudar o plano se necessário.

Essa flexibilidade poder ser desconcertante para gerentes de produtos, mas é uma parte necessária do processo de construir ótimos produtos. Então, esteja confortável quanto a ambiguidades. Há muito delas neste trabalho.

2.3 Para ser justo...

Terminarei esta seção com uma característica bônus. Aliás, é a característica mais importante de um gerente de produtos — aquela que rege todas as outras.

Uma vez tive uma discussão com um colega no nosso time de desenvolvimento sobre o novo processo de desenvolvimento que tínhamos lançado uns meses antes. Uma das palavras que ele usou para descrever o novo processo era "justo".

Foi um comentário passageiro e eu não pensei realmente muito nisto no momento, mas desde então continuo voltando àquela conversa e à importância da justiça na profissão de gestão de produtos. Todas as características das quais acabei de falar são ótimas, mas sobretudo, justiça é aquela que um gerente de produtos não pode deixar de ter.

Vejamos a definição da palavra "justo" (Definição de *fair* pelo Free Online Dictionary, Thesaurus and Encyclopedia, em http://www.thefreedictionary.com/fair):

> **JUSTO.** adj. livre de favoritismos ou interesse pessoal ou tendências ou fraude.

Sem favoritismo

Uma das maneiras mais rápidas de se tornar ineficaz em uma organização de gestão de produtos é começar a favorecer um grupo interno em particular, uma linha de produto, ou base de usuários. Tão logo as pessoas sentem que você não está olhando para todas as ideias e sugestões com justiça e igualdade, segue-se inevitavelmente uma falta de confiança. E sem confiança, você terá de trabalhar muito mais (e por mais tempo) para conduzir as pessoas pelo trajeto do seu roteiro.

Sem interesses pessoais

Se você começar a fazer coisas com motivos puramente como "porque eu quero" ou "porque eu estou sendo medido por esta métrica", aquela mesma falta de confiança cresce muito rapidamente. Você não pode ser eficaz se fica nutrindo seus próprios projetos pessoais e ignorando todas as outras necessidades ao seu redor.

Sem tendências

Isso frequentemente acontece quando gerentes de produtos recebem notícias que não querem ouvir, especialmente da equipe de analytics ou de pesquisa de usuário. Se alguma coisa não vai bem no teste, não invente razões para você estar certo e os usuários errados. Faça a coisa certa e realinhe o design.

Uma das habilidades mais difíceis que um gerente de produtos deve aprender é tirar seus próprios sentimentos e emoções fora da equação, quando se trata de tomadas de decisão. Sim, vai uma boa parte de instinto e pressentimento na visão de produto, mas isso não pode se basear em preferências pessoais ou ideias preconcebidas. É muito mais fácil falar do que fazer, mas é algo para o qual se esforçar e estar atento a todo o momento.

Sem fraudes

Por mais que pareça óbvio, você ainda vê isso frequentemente, especialmente quando se trata de métricas e avaliações. Não ignore ou distorça dados negativos, nem diga que o problema é por culpa de outra pessoa. É tarefa do gerente de produtos ter domínio sobre um produto — e isso quer dizer tanto seus sucessos quanto falhas. Você ganhará confiança e respeito se não apenas reivindicar os sucessos, mas também reconhecer as falhas e empenhar-se a fazer melhor da próxima vez.

Com frequência, o papel do gerente de produtos é referido como "o grande diplomata", e com motivo. É nossa responsabilidade equilibrar a variedade de necessidades de dentro e de fora do

negócio, e de alguma forma transformar isso em um *roadmap* que entregue valor de negócio, bem como atenda às necessidades dos usuários. É mantendo o foco na justiça que se alcança este objetivo.

- **Ser justo com os usuários —** Abordar usuários com respeito, abertura e transparência. Entender suas necessidades e explicar-lhes quando você está fazendo algo que torna mais difícil alcançar estas necessidades.
- **Ser justo com o negócio —** Faça tudo o que puder para entender as necessidades do mercado, comercialização, suporte ao consumidor e tudo o mais. Coloque-os no processo de planejamento, seja claro sobre como projetos são priorizados, e ajude-os a ajustarem-se àquele processo, de modo que possam definir as metas do projeto do jeito certo para colocá-las no roadmap.
- **Ser justo com a tecnologia —** Não force o time de desenvolvimento a fazer a tecnologia executar algo que não é capaz. Entender a dívida técnica na organização e trabalhar ativamente para tornar essas melhorias parte dos ciclos de desenvolvimento regulares.

Muito disso vem naturalmente em bons gerentes de produtos, mas devemos tomar consciência disso todos os dias. Justiça é o mínimo exigido de um gerente de produtos eficaz. Se você fizer certo, o processo real de desenvolver ótimos produtos pode começar. Mas se não, você está estagnado, trabalhando com um time que não tem motivo para confiar se você está fazendo a coisa certa.

2.4 Onde gestão de produtos se encaixa em uma empresa

Eis algumas más noticias. Não importa quão bem gerentes de produtos tenham as características ideais para a função; se eles são deixados de lado em decisões cruciais, eles não serão eficazes. Então, outra pergunta comum sobre gestão de produtos é onde ela deve se encaixar na hierarquia organizacional. Gerentes de produtos são frequentemente descritos como mini-CEOs de seus produtos, o que devia servir como um bom indicativo do nível hierárquico apropriado que eles requerem para operar com eficácia.

Gestão de produtos não pode ficar junto com o marketing ou engenheiros — o que, infelizmente, às vezes acontece porque empresas não têm certeza de onde colocá-los. Para ser eficaz, a gestão de produtos precisa de representatividade no nível executivo.

Em startups com um ou dois gerentes, isso significa que eles reportam diretamente ao CEO ou COO. Em empresas maiores, isso quer dizer que a equipe de gestão de produtos é guiada por uma pessoa de nível executivo. Nos últimos anos, isso tem evoluído para um novo cargo, o CPO (*Chief Product Officer*).

Entretanto, rótulos e nomes de cargos não são tão importantes quanto o que isso implica: que o time de gestão de produtos colabora com seus colegas tomadores de decisões na empresa. Esconder gerentes de produtos em uma empresa com metas e agendas muito específicas (como marketing ou transações) destrói o propósito de desenvolver e executar uma estratégia e visão de produto holísticas.

Não gosto muito de falar sobre cargos e hierarquia, então não leve isso como uma propaganda pessoal para elevar os gerentes de produtos a um status divino. Em vez disso, conforme você verá por meio deste livro, eles devem estar em um nível de tomadas de decisão — qualquer nome que isso tenha em sua empresa, e tenha dez ou dez mil funcionários —, para que possam fazer os melhores produtos possíveis para a empresa e seus usuários. E isso é tudo.

2.5 Um pré-requisito para o sucesso

Há um último tópico que precisamos apontar antes de entrar nos detalhes do que gerentes de produtos fazem no dia a dia. Uma empresa pode contratar os melhores gerentes do mundo, e implementar os melhores processos de desenvolvimento, mas ainda falhará se algum pré-requisito crucial para o sucesso não for cumprido. Este pré-requisito é:

> Para os gerentes de produtos terem sucesso, deve haver um mandato executivo e um entendimento pela empresa toda de que, mesmo que todos tenham uma voz, as decisões de produtos ficam com os gerentes de produtos em última análise.

Isso é algo difícil de engolir para muitas empresas — e é outra razão por que acabei de argumentar que eles precisam ter uma representatividade no nível executivo. Quando eu falo isso em cursos de treinamento de gestão de produtos, o clima da sala frequentemente muda. É aí que as pessoas começam a reclamar que, mesmo que eles vejam o valor da função de gerente de produtos, isso nunca funcionaria em suas empresas, porque os líderes não estão dispostos a abrir mão deste controle final sobre o produto.

Nós discutiremos estratégias para lidar com isso durante o livro, mas por enquanto, eis uma lembrança do que Seth Godin declaradamente disse uma vez: *"Nada acontece quando todo mundo tem de concordar".* O gerente de produtos está lá para assegurar que coisas aconteçam — as coisas certas.

Times executivos e contribuidores individuais têm de comprar este papel. Se não, gerentes de produtos se tornam espectadores impotentes e frustrados de um processo que continua uma espiral fora de controle. E eles terminarão indo a algum lugar onde são apreciados pelo valor que trazem.

2.6 A seguir...

Neste capítulo, cobrimos vários dos que devem ser considerados assuntos leves em gestão de produto: o que são os gerentes de produtos, como eles trabalham com outras pessoas, o que diferencia os bons dos ruins.

É tentador descascar esses assuntos para chegar logo no como: os processos e as atividades que compõem o dia a dia do papel de um gerente de produtos. Mas isso seria um erro. Eu nunca vi uma função no desenvolvimento de produto que demanda mais desse tipo de habilidades sociais para seu sucesso do que o de gerente de produtos.

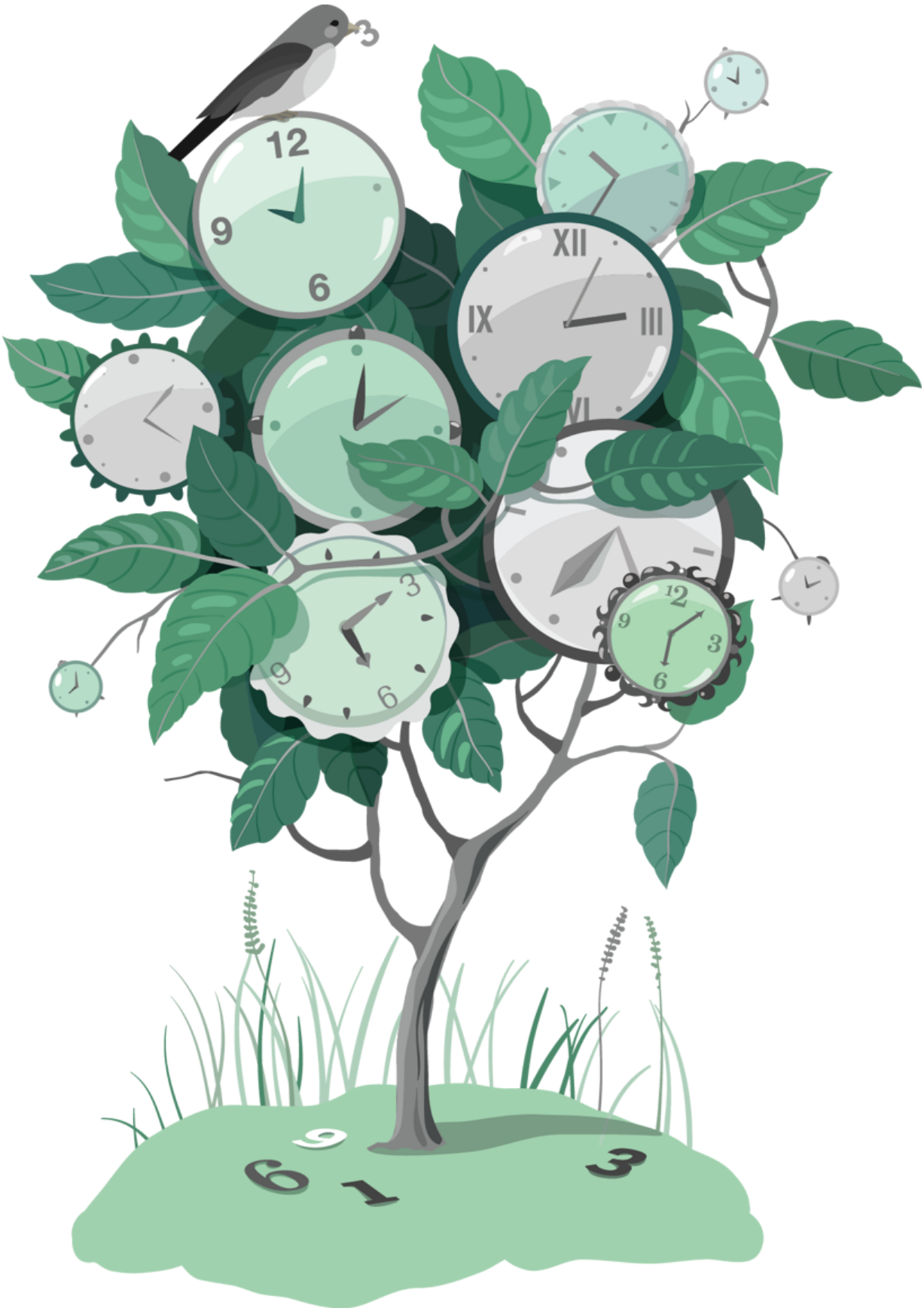
Um gerente pode ter a melhor estratégia, e ser brilhante na execução, mas sem a habilidade de trabalhar bem com pessoas e conseguir que elas se dediquem a uma causa comum, ele vai falhar. Então, se você pulou uma das seções deste capítulo, agora é uma boa hora para voltar e lê-las com carinho.

Agora que estabelecemos esta base, é hora de seguirmos em frente e vermos como os gerentes de produtos passam seus dias. Vamos dividir o conteúdo em duas partes: planejamento de produto (como preparar e priorizar mudanças no produto); e execução do produto (como encaminhar estas mudanças).

Referências

DAY, George S.; NEDUNGADI, Prakash. Managerial representations of competitive advantage. *Journal of Marketing 58*, abr. 1994, p. 40.

Parte 2 — Planejamento


12
9
6
XII
IX
III
VI
3
9
6
12
9
3
6
9
6
1
3

Capítulo 3
Descobrindo necessidades

"Se eu tivesse perguntado às pessoas o que elas querem, eles teriam dito 'cavalos mais rápidos'". É muito comum estas palavras (supostamente ditas por Henry Ford) serem usadas como uma maneira de justificar uma corrida precipitada para executar uma tal inovação antes de a ideia ser testada com os usuários.

Vale a pena notar que, não apenas Henry Ford provavelmente nunca falou tais palavras, mas também acabou que esse tipo de pensamento resultou em "uma perda catastrófica de quota de mercado da qual [a Ford] nunca se recuperou (*Henry Ford, innovation, and that 'faster horse' quote*, http://smashed.by/henry-ford).

A lição que devemos aprender desta história é que é extremamente perigoso executar ideias sem primeiro identificar e testar as hipóteses sobre seus valores. Não deveríamos pular para uma solução antes mesmo de entender o problema. E é disso que este capítulo se trata.

Eu já fiz alusão aos dois principais elementos da gestão de produtos sobre os quais vamos discutir: planejamento do produto e execução do produto. Agora é a hora de expandir sobre a parte do planejamento do produto. A seguir, está um diagrama do framework que usaremos para discutir as atividades de planejamento pelas quais gerentes de produtos são responsáveis.

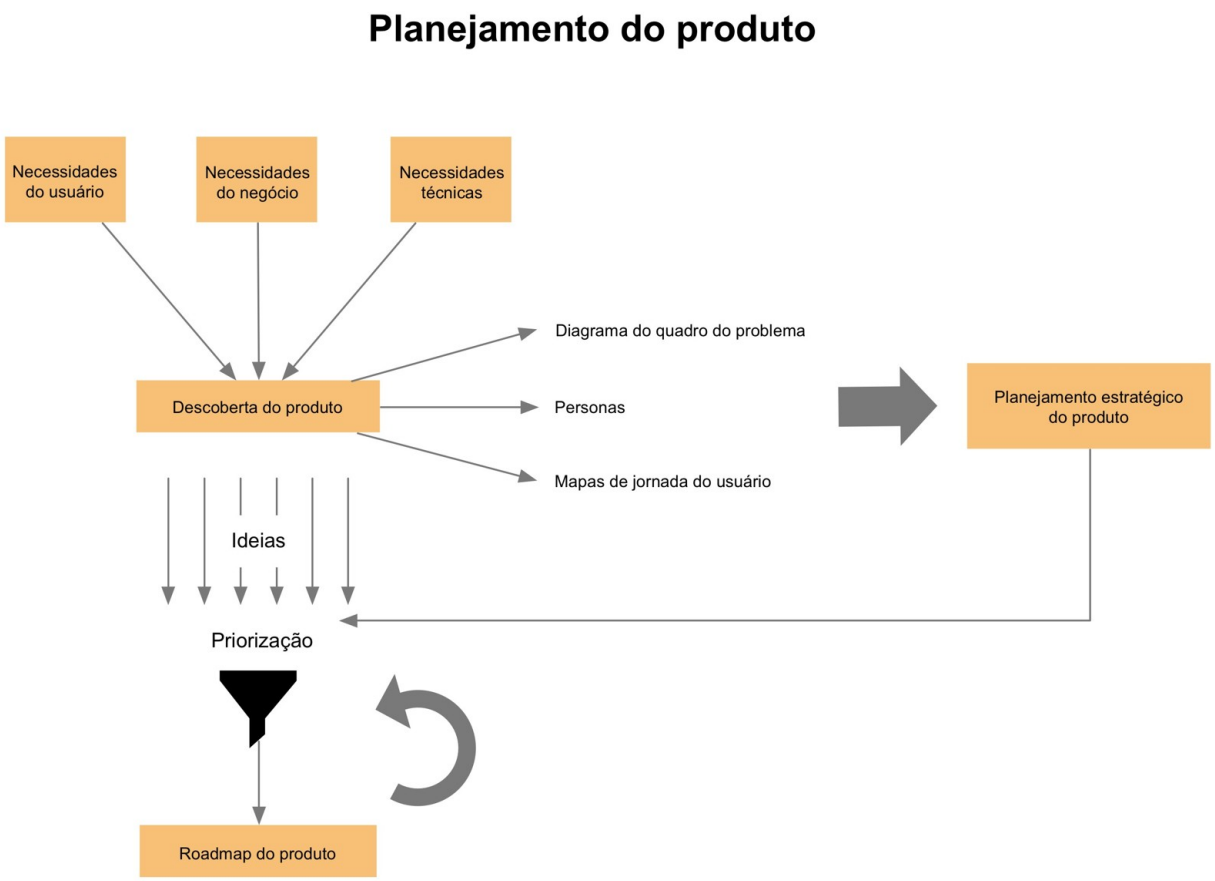


Figura 3.1: Os primeiros passos para o processo de planejamento do produto, desde identificar necessidades até desenvolver um roadmap estratégico e flexível

Começaremos olhando os diferentes fatores do processo de desenvolvimento. O ponto de partida são — sempre — **necessidades**. Não é o que nós achamos que seria legal, mas sim o que os usuários ou o negócio precisam para serem bem-sucedidos. Alguns fatores neste processo incluem:

- **Necessidades do usuário —** Os gerentes de produtos devem ter um bom entendimento do mercado, dos clientes da empresa (existentes e em potencial), e seus comportamentos e atitudes. Eles não devem nunca ser pegos de surpresa por questões sobre o público-alvo do produto. Vamos ver diferentes fontes dos dados de usuário, incluindo pesquisa de mercado, de

experiência do usuário (XP), *analytics* do site e suporte ao consumidor.

- **Necessidades do negócio —** O mantra de "colocar usuários em primeiro lugar" muito frequentemente é negligente com o fato de que um produto existe para fazer dinheiro. Porém, ter uma meta de renda não é uma desculpa para design ruim, portanto, vamos ver a diferença entre fluxos de receita bons e ruins.
- **Necessidades técnicas —** As necessidades do desenvolvimento, durante muito tempo, são ignoradas em detrimento dos requerimentos mais tangíveis de front-end e de negócio. Desenvolvedores sabem as limitações do produto; eles sabem o que precisa ser consertado, e quais demandas técnicas devem ser cumpridas. Vamos discutir os mecanismos do tão importante relacionamento entre desenvolvedores e gerentes de produtos.

Todas essas diferentes necessidades alimentam um processo chamado "a descoberta do produto". Mais uma vez, há diferentes definições para este processo, mas aqui usarei esta expressão para me referir a:

> *Definir o problema que você está tentando resolver para o usuário, as oportunidades de negócio que existem para resolver os problemas, e as competências centrais que vão ajudá-lo a fazer com que a solução seja um sucesso.*

Os resultados do processo de descoberta do produto podem variar — assim como o tempo necessário pode variar entre duas ou três horas até várias semanas. Em geral, o processo de descoberta de produto em projetos grandes resulta em coisas como diagramas para enquadrar problemas, personas, e mapas da jornada do cliente — vamos discutir tudo isso em detalhes. Esses artefatos compõem

o plano estratégico do produto: um documento que resume o que é o produto, para quem, e os planos para torná-lo um sucesso.

Uma vez que a estratégia está posta, o gerente guia um processo de geração de ideia (acompanhado de várias abordagens diferentes para resolver um produto) e iteração (rapidamente filtrar essas ideias, deixando apenas as melhores). Isso é seguido pela validação do cliente (testar ideias com o público-alvo) como parte de um processo maior de priorizar quais ideias valem a pena seguir.

Todas essas atividades fazem parte do poderoso roadmap do produto. Há bastante controvérsia sobre o valor e a legitimidade dos roadmaps, então discutiremos isso em detalhes (*spoiler:* não é de todo ruim).

Uma vez que o planejamento estratégico do produto e o roadmap inicial estão prontos, a execução pode começar. Agora, provavelmente você está tentado a pular esta seção e pular diretamente para a execução, mas não faça isso! Um dos maiores perigos do desenvolvimento de produto é pular para a execução antes de um ciclo de planejamento apropriado ser feito, então precisamos dar ao planejamento a atenção que ele merece.

Vamos começar agrupando as necessidades do usuário.

3.1 Necessidades do usuário

A primeira coisa que precisamos esclarecer é a diferença entre necessidades e funcionalidades. Frequentemente, cometemos o erro de equiparar as features do produto com as necessidades do usuário.

Se você já usou um eletrodoméstico, você sabe que não é bem assim. Você já usou mais do que um ou dois dos ciclos pré-setados da sua máquina de lavar roupa? E de quantas maneiras diferentes

você precisa tostar seu pão? A evolução dos eletrodomésticos é um exemplo perfeito do que acontece quando funcionalidades correspondem ao valor. Não precisamos de mais jeitos de lavar as roupas. Precisamos de maneiras mais rápidas e silenciosas, claro.

Mas, como sabemos, mais não é necessariamente melhor. E é aí que os usuários às vezes precisam agir por conta própria.

TV ON
&TV POWER
ON
OFF
TV OFF
CHANNELS
CHAN
PAGE
1
2abc
3def
4ghi
5jkl
6mno
7pqrs
8tuv
9wxyz
DASH
0
ENTER
DIRECTV
SOUND ON
VOLUME
+
−

Figura 3.2: Fonte: Reddit “My buddy dad-proofing his remotes”, em http://smashed.by/remotes

Quando as primeiras reviews e estatísticas de uso da homepage do Facebook começaram a aparecer (*The Facebook home disaster*, http://smashed.by/fb-home), John Gruber usou uma frase que ficou na minha cabeça (*Facebook home is looking like a flop*, http://smashed.by/fbflop): *"É uma implementação bem planejada de uma ideia que ninguém quer".*

Deixando os exageros de lado, é isso que acontece quando features (feed de notícias, amigos por toda a tela, mensagens de bate-papo, carregador de apps etc.) são confundidas com necessidades do usuário (por que as pessoas iriam querer substituir o sistema operacional do seu celular por um app?). A distinção entre features e necessidades é importante, e às vezes difícil de apontar. É onde entra a pesquisa de usuário.

Métodos de pesquisa para agrupar necessidades dos usuários são poderosos porque se baseiam mais em observação e dedução do que reunir respostas para um apanhado de perguntas predeterminadas. Mas antes de entrarmos nesses diferentes métodos que podemos usar para fazer melhores produtos, precisamos fazer um pequeno tour para definir alguns termos básicos de pesquisa.

Primeiro, precisamos distinguir pesquisa quantitativa de pesquisa qualitativa. Com abordagens quantitativas, os dados tendem a ser coletados indiretamente da pessoa questionada, por meio de métodos como questionários e web analytics.

Pesquisa quantitativa permite que você entenda o que está acontecendo, ou quanto disso está acontecendo. Com abordagens qualitativas, os dados são coletados diretamente dos participantes, por meio de entrevistas ou testes de usabilidade. Pesquisa qualitativa o ajuda a entender como ou por que certos comportamentos ocorrem.

Também é preciso fazer uma distinção entre pesquisa de mercado e pesquisa de usuário. Ambas são importantes, mas cada uma serve para propósitos diferentes. Pesquisa de mercado busca compreender as necessidades de um mercado em geral. Ela preocupa-se com coisas como equidade de marcas (*brand equity*) e posicionamento de mercado.

Levantamento de atitudes e grupos de foco são o pão-com-manteiga das ferramentas para pesquisadores de mercado. Elas têm a missão de descobrir como posicionar um produto no mercado. Pesquisas e grupos de foco são muito úteis para entender tendências e necessidades do mercado, mas eles não o ajudarão a entender muito quando se trata de projetar seu produto.

Pesquisa de usuário, por outro lado, foca em interações entre o usuário e o produto. Ela se preocupa com como as pessoas interagem com tecnologia, e o que nós podemos aprender com seus desejos, suas necessidades e frustrações. É nesses métodos que focaremos nesta seção.

Então, com essas definições em mãos, vamos ver alguns dos métodos mais comuns de pesquisa de usuário disponíveis. No geral, podemos classificar métodos de pesquisa em três diferentes pilhas. Todos os métodos que eu mencionar aqui (e mais) podem ser vistos em detalhes no livro *Observing the User Experience*, de Elizabeth Goodman, Mike Kuniavsky e Andrea Moed.

Pesquisa de exploração

A pesquisa de exploração é útil principalmente quando o objetivo é descobrir as necessidades mais importantes (e frequentemente pendentes) que usuários têm com os produtos e serviços ao seu redor. Isso inclui métodos como investigações contextuais (também chamados de pesquisa etnográfica ou visita de campo), sessões de design participativo e teste de conceitos.

O objetivo aqui é descobrir onde estão essas lacunas no modo de como produtos existentes estão resolvendo os problemas dos usuários. Frequentemente, um novo produto ou ideias de features aparecem nessas sessões.

Não se confunda. Não se trata de perguntar às pessoas se elas querem cavalos mais rápidos, trata-se de observar pessoas e descobrir que elas precisam chegar aonde querem muito mais rápido do que atualmente podem.

Por exemplo, costumávamos fazer bastantes pesquisas etnográficas com vendedores do eBay por todo o mundo. Indo às casas das pessoas e vendo como elas gerenciavam suas vendas, descobrimos um assunto maior, que as web analytics e pesquisas nunca poderiam nos contar. Os vendedores têm jeitos particulares de gerenciar suas vendas, desde recados em post-its grudados em seus monitores até planilhas no Excel com fórmulas complicadas e tabelas dinâmicas.

Os vendedores são forçados a criar seus próprios processos, em algo que o eBay os deveria estar ajudando: como acompanhar o progresso das vendas e aprender com isso. Por meio da etnografia, descobrimos uma necessidade de usuário não atendida, que poderia ser resolvida com uma variedade de features no site. Mas a necessidade é o ponto de partida.

Pesquisa de design

A pesquisa de design ajuda a desenvolver e a aperfeiçoar as ideias do produto que surgem das análises sobre as necessidades dos usuários. Os métodos incluem o tradicional teste de usabilidade, *RITE testing* (sigla em inglês para *rapid iterative testing and evaluation*, isto é, teste rápido e iterativo e avaliação), e também métodos quantitativos, como *eye tracking*.

Essa classe de pesquisa nos ajuda, durante o processo de design, a criar melhores produtos para os problemas dos usuários que

estamos tentando resolver. Por exemplo, nós podemos construir protótipos iterativos e trazer as pessoas para o laboratório de usabilidade, dar a elas tarefas para completar nesse protótipo, e descobrir problemas de usabilidade antes de começarmos o (caro) ciclo de desenvolvimento. Desde que estes são normalmente profundas entrevistas individuais, é também uma grande oportunidade de conseguir mais insights sobre quão bem certas features encontram as necessidades dos seus usuários, que identificamos durante a pesquisa explanatória.

Pesquisa de avaliação

A pesquisa de avaliação nos ajuda a descobrir se as mudanças que fizemos realmente melhoraram o produto, ou se elas não nos levaram a lugar nenhum. Esta categoria de pesquisa é frequentemente negligenciada, porém é uma parte crucial do ciclo de desenvolvimento do produto.

Dentre os métodos, estão pesquisas e web analytics para nos fornecerem uma visão da performance do nosso produto ao longo do tempo, não apenas em termos de conversões rígidas, mas também em termos de atitudes dos usuários. Esses métodos são úteis principalmente quando combinados com pesquisas de design mais a fundo para entender por que estamos vendo tais mudanças.

Por exemplo, formulários de analytics podem nos dizer qual o ponto onde as pessoas abandonam um formulário. Uma vez que fizermos as melhorias de usabilidade para o formulário, é importante avaliar se tais alterações fizeram a diferença nas taxas de conclusão. Sem pesquisa de avaliação, não saberemos se estamos indo na direção certa.

Então, com esse framework em mente, uma das primeiras coisas que um gerente de produtos precisa fazer é encontrar todas as pesquisas existentes que possam ajudar a descobrir necessidades do usuário — tanto gerais (o que o produto precisa fazer para ajudar o usuário a alcançar seus objetivos) quanto específicos (o que está

quebrando ou faltando na solução atual). Junte toda e qualquer pesquisa, relatórios de teste de usabilidade e de web analytics que você puder, e beba dessa fonte.

O que os usuários realmente querem e necessitam de um produto deve se tornar parte da formação do gerente de produtos, e a única forma de alcançar isso é mergulhar nos dados do usuário. Faça o que for necessário: marque ligações semanais ao consumidor, trabalhe na fila de suporte ao consumidor algumas horas por semana, ou prepare sessões regulares de teste de usabilidade.

Usuários devem ter uma voz constante nos ouvidos do gerente quando eles começam a balancear todas as outras demandas de negócio e de tecnologia, que estão sempre fazendo pressão — especialmente uma vez que essas outras demandas entram em competição com as necessidades do usuário com frequência.

3.2 Necessidades de negócio

A web está cheia de restos de produtos que foram bons em cumprir necessidades dos usuários, mas que nunca encontraram um jeito de fazer dinheiro e se tornar um negócio sustentável. Nos últimos anos, vimos vários serviços queridos na web se desligarem por falta de receita. A Editorially era uma ferramenta colaborativa fantástica de escrita e edição, cujos criadores eventualmente perceberam que "Mesmo que todos os nossos usuários pagassem, ainda não seria o suficiente (*Goodbye*, http://smashed.by/editorially-goodbye).

Alguns meses antes disso, a startup de foto Everpix fechou suas portas, parcialmente porque não conseguia bancar as contas de armazenamento em nuvem. Isso aconteceu mesmo tendo milhares de usuários pagantes na plataforma. Os fundadores depois admitiram que, mesmo que eles pudessem fazer um produto que as pessoas genuinamente amassem, eles gastavam muito tempo

trabalhando naquele produto, mas não tempo suficiente investindo no crescimento e distribuição (*Out of the picture: why the world's best photo startup is going out of business,* http://smashed.by/photo-startup).

Esses casos são complexos e nunca há respostas fáceis. Ainda assim, é uma abordagem comum na web focar em ganhar o máximo de usuários possível o mais rápido possível, e então pensar em como fazer dinheiro com isso depois. Você pode me considerar *old school*, mas não é assim que se constrói um negócio.

Não estou dizendo que um novo produto deve ser lucrativo desde o primeiro dia (embora isso fosse bom, é claro). Mas é preciso pelo menos ter um plano — alguns fluxos possíveis de renda — que eventualmente levariam a um negócio sustentável. A menos que você esteja apenas a fim de *acqui-hire* — isto é, o processo de adquirir uma empresa para recrutar seus funcionários —, mas essa é uma história completamente diferente.

Então de onde vêm as ideias de fluxos de receita? Bem, em muitos casos, elas vêm dos consumidores. Os métodos de pesquisa que discutimos anteriormente também podem ser usados para descobrir aquilo pelo qual as pessoas estariam dispostas a pagar — e quanto.

Mas também existem vários times que passam a maior parte do tempo pensando em necessidades do negócio, e o gerente de produtos deve ser um forte aliado deles. Isso inclui os times de desenvolvimento, de vendas e de marketing, e o time de engenheiros (sim, o time de engenheiros — ninguém conhece o produto melhor do que eles).

Quando se trata de fazer o negócio crescer, é útil separar as atividades em duas filas: eliminar maus fluxos de receita, e buscar bons fluxos de receita.

Eliminando maus fluxos de receita

Sófocles, escritor de tragédias gregas, disse *"Lucro é doce, mesmo se resulta de um engano"*. Temos de estar cientes de nossas próprias fragilidades quando se trata de fazer dinheiro. Enganar pessoas para fazer uma grana fácil pode parecer uma boa ideia no momento, mas é uma estratégia para curto prazo, destinada a sair pela culatra — sem falar que isso não se encaixa exatamente nas características éticas que discutimos anteriormente.

No design de interface, nós nos referimos a técnicas fraudulentas como padrões escuros. Um padrão escuro é um tipo de interface especialmente projetada para enganar as pessoas para elas comprarem coisas que não querem. Existem muitos exemplos, e o site Dark Patterns (http://www.darkpatterns.org/) fornece uma lista abrangente. Mas essas técnicas incluem exemplos como:

- A Ryanair deixa a opção de remover o seguro de viagem escondida em um menu dropdown descontextualizado, de modo que a maioria das pessoas não percebe que está comprando o seguro.
- Alguns aplicativos iOS para crianças, como o Meu Talking Tom, colocam pop-ups aleatórios na tela em todos os pontos do jogo, para fazer as crianças a realizarem compras internas na app.
- Na tela de login, o PayPal frequentemente mostra uma propaganda de tela cheia, com apenas um pequeno link no canto direito para fechá-la e seguir para a tela de sua conta.
- O jogo FarmVille, da empresa Zynga, foi "projetado com um objetivo em mente: coagir usuários a cuidar de seus terrenos virtuais o maior tempo possível" (*The Zynga abyss*, http://smashed.by/zynga-abyss).

Pode parecer óbvio apontar que algumas fontes de renda não são éticas, portanto, não vale a pena ir atrás delas. O problema é que, muito frequentemente, esses métodos funcionam — eles infelizmente fazem dinheiro, sim (pelo menos, em curto prazo). Mas

eles também têm consequências em longo prazo, que raramente são levadas em conta.

Uma vez que os usuários entendem o que está acontecendo e começam a reclamar, esses métodos clandestinos afetam as empresas diretamente, na forma de custos elevados para suporte e grandes danos de reputação. A Ryanair acabou virando garoto-propaganda dos padrões escuros devido às suas táticas. Essa não é uma boa posição para se estar.

O negócio é que pouquíssimas pessoas começam seus dias pensando "Quantas pessoas será que eu poderia enganar hoje?". Em vez disso, os padrões escuros e métodos enganosos se esgueiram até os gerentes de produtos, apresentando-se como ideias potencialmente decentes que vão se degenerando pouco a pouco, até tornarem-se padrões escuros.

Nem devíamos gastar tempo falando disso, exceto para dizer: cuidado. Não caia nessas armadilhas de padrões escuros. Uma regra de ouro fácil, óbvia, mas raramente aplicada, é questionar para cada oportunidade de renda em potencial: "Eu estaria ok se um produto me pedisse para fazer ou pagar por isso?".

Se a resposta é "não", caia fora. Há caminhos melhores. Pode ser mais difícil encontrá-los, mas vale a pena sacrificar sucesso de curto prazo por fidelidade de clientes em longo prazo. Além disso, você dormirá melhor de noite.

Às vezes, um fluxo de renda começa como uma boa ideia, mas uma mudança no ambiente externo o faz virar algo indesejável. O problema é que, até lá, isso já deve ter se tornado uma larga fonte de renda, o que deixa a empresa em um dilema bem difícil.

Um exemplo disso são as fotos nos resultados de busca no eBay. Lá em 1995, quando o eBay foi fundado, armazenamento era caro. Então fazia sentido cobrar os usuários uma quantia nominal para subir uma foto em suas listagens. Uma década depois, em 2005, não apenas armazenamento era barato, como também a ideia de

cobrar usuários para adicionar fotos às listagens era ridícula. O problema era que, até então, cobrar por fotos era uma fonte de renda significante, logo, não foi uma decisão fácil tornar as fotos gratuitas.

Nossa equipe de experiência de usuário se juntou com a de analytics para mostrar que exibir fotos em resultados de busca por padrão não apenas aumentava as vendas como também tinha um impacto positivo no ranqueamento de organização e prestatividade dos resultados de busca. Levou um tempo, mas eventualmente o eBay tomou a decisão corajosa de terminar aquele mau fluxo de renda e tonar as fotos gratuitas (até oito fotos), e não olharam para trás.

Quando se trata de lidar com esses maus fluxos de renda acidentais, a melhor ação a tomar é dirigir pesquisas para entender as necessidades e motivações, juntamente com testes A/B para ter uma medida precisa do efeito que isso terá na receita boa, quando a má fonte de renda for cortada.

Buscando bons fluxos de renda

Bons fluxos de renda podem vir de diferentes fontes. Consumidores estão dispostos a pagar por coisas desde que o valor esteja imediatamente claro para eles. E o processo de gestão de produtos inteiro é construído com base em encontrar este valor primeiro, e então construir um produto e um negócio em torno dele, em vez de criar algo e então sair procurando o valor. Portanto, a pesquisa de necessidades dos usuários é sempre o primeiro lugar para procurar como o produto pode fazer dinheiro.

Vejamos o talvez um pouco excêntrico exemplo do Iron Maiden (sim, é verdade). A banda gastou uma quantidade significante de tempo em turnê pela América do Sul, e isso acabou sendo uma ótima estratégia. Eles costumam tocar em shows esgotados, altamente lucrativos nesta região.

Por que a banda faz tanto sucesso aqui? Se você perguntar para a empresa de *music analytics*, Musicmetric, eles dirão que tem a ver com um fenômeno improvável: pirataria. A empresa detectou uma onda de tráfego no BitTorrent relacionado à música do Iron Maiden na América do Sul nos últimos anos, principalmente no Brasil.

Ir aonde as pessoas já são fãs da música possibilitou a banda, como o CEO da Musicmetric Gregory Mead afirma, a "ser muito bem-sucedida em transformar compartilhamento gratuito de arquivos em público pagante" (*How Iron Maiden turned piracy into paying customers*, http://smashed.by/piracy). Entender onde seus usuários já estão investindo em seu produto é a melhor forma de descobrir aquilo pelo qual eles estariam dispostos a pagar.

Uma vez que existem fluxos de renda, há várias estratégias de crescimento padrões, tais como expandir para novas regiões, estabelecer novos canais, atrair mercados maiores, e construir novos produtos para o mercado existente. Mas há uma estratégia em particular em que eu gostaria de focar, porque é uma abordagem ampla para o sucesso de negócios em longo prazo. É uma estratégia formalizada por Brandon Schauer, na Adaptive Path, e é chamada *O longo Uau* (http://smashed.by/long-wow). Parafraseando o artigo de Brandon sobre isso:

> *O longo **UA** é um meio de alcançar fidelidade de clientes em longo prazo, por meio de sistematicamente impressioná-los mais e mais. Estando um passo à frente de ser apenas uma medição de fidelidade, o longo Uau é uma abordagem experimental para nutri-la e criá-la.*

O longo Uau se desenvolve em um processo de quatro etapas:

1. **Conheça sua plataforma de entrega —** Identifique os jeitos de combinar diferentes formas de engajar consumidores, tanto online como offline.

2. **Ataque uma área grande de necessidades de usuário não atendidas —** Com base em suas pesquisas de necessidade de usuário, identifique uma área onde há uma enorme demanda que seu produto e outros lá fora não atenderam ainda.
3. **Crie e evolua seus processos repetitivos —** Combine o poder atual de sua empresa com novas ideias para encontrar necessidades não atendidas, para desenvolver caminhos de surpreender seus usuários mais e mais vezes.
4. **Planeje e organize experiências Uau —** Desenvolva suas ideias com o tempo, e introduza experiências novas e melhores, consistentemente, ao longo do ciclo de vida de desenvolvimento do produto.

Então, repita o quanto for necessário para assegurar-se de que o longo Uau não é apenas coisa de uma única vez. Esta é uma excelente maneira de identificar boas fontes de renda em seu produto, e garantir que você continuará a fornecer o valor necessário para criar uma clientela fiel, que continuará a pagar para você.

3.3 Necessidades técnicas

A primeira coisa que precisamos esclarecer quando se trata de necessidades técnicas é que, assim como em finanças, existe uma grande diferença entre ativos e dívidas. Ativos técnicos são coisas como tecnologias adjacentes nas quais seu produto é construído, sistemas de retaguarda (procuradoria, finança, fulfillment) e escalabilidade de tecnologias (atividades de administração de sistema).

Em contraste, dívida técnica refere-se a sistemas e código que fazem tensão no produto (na forma de bugs e problemas de escalabilidade), e fica pior enquanto não for resolvido. De acordo

com Steve McConnell (*Technical debt*, http://smashed.by/technical-debt), existem dois tipos de dívidas técnicas:

- **Dívidas não intencionais** ocorrem quando o design técnico errado é implementado, ou mesmo quando um programador escreve um código ruim. Este tipo de dívida é não estratégica, e você quer tê-las o mínimo possível.
- **Dívidas intencionais** ocorrem quando a empresa sabe que o que estão fazendo não é o ideal, mas é um compromisso que vale a pena ser feito, por quaisquer motivos (geralmente tem a ver com orçamento ou prazo). Embora não seja ideal, este tipo de dívida é inevitável em qualquer empresa. É preciso lidar com ela e minimizá-la, mas diferente de dívidas não intencionais, não é completamente inevitável.

Há muitos motivos óbvios para evitar, minimizar e amortizar a dívida técnica. Mas o argumento mais saliente é evitar o que é comumente referido como a teoria da janela quebrada (http://smashed.by/broken-windows). Essa teoria criminal busca explicar o efeito do caos e vandalismo urbanos, e afirma que:

> *A manutenção e monitoramento dos ambientes urbanos em uma condição bem ordenada pode brecar vandalismos mais graves e o agravamento para crimes mais sérios.*

A analogia básica é que um software é como um ambiente urbano. Tão logo que aparecem algumas janelas quebradas (código ruim), e estas não são imediatamente consertadas, a tendência é que vândalos quebrem mais janelas (nós deixamos de nos preocupar em fazer bom código). Então, o ambiente começa a deteriorar: aparecem lixos, mendigos e por aí vai (todos os padrões de código vão pela janela). Em pouco tempo, o inferno inteiro está aí.

Para citar Steve McConnell mais uma vez (*Technical debt*, http://smashed.by/technical-debt):

> *Se a dívida cresce muito, eventualmente a empresa vai gastar mais em servir esta dívida do que investir em aumentar o valor dos seus outros ativos. Um exemplo comum é um banco de código legado, no qual muito trabalho é exigido para manter um sistema de produção rodando (por exemplo, "servindo a dívida"), que sobra pouco tempo para acrescentar novas capacidades ao sistema.*

Essa é uma situação que deve ser evitada a todo custo, e é aí que entra o gerente de produtos. Encontrar espaço no roadmap para resolver uma dívida técnica costuma ser uma tarefa extremamente difícil de vender. Pagar dívidas não é sexy — você geralmente não consegue ver mudanças no front-end, e pouquíssimas pessoas entendem o que está acontecendo.

Então, há um pouco de ceticismo sobre por que esse trabalho é necessário, e ninguém realmente quer ir lá e limpar as sujeiras no código. Mas é essencial priorizar dívida técnica na maior parte, se não em todos, dos ciclos de desenvolvimento, para evitar um colapso de serviços básicos até que não sobre nada além de uma cidade fantasma.

É importante notar que dívida técnica não é necessariamente ruim no momento em que ocorre. Às vezes, ela permite um lançamento de uma feature rica, enquanto, de outro modo, não aconteceria dentro de uma abordagem de tolerância zero contra dívida técnica. Em geral, novas dívidas são ok, dívidas antigas são ruins.

Henrik Kniberg propõe uma ótima forma de lidar com dívidas técnicas fora de controle, no artigo *Good and bad technical debt* (http://smashed.by/good-bad-debt). Ele apresenta o conceito de um teto da dívida, a partir do qual você não tem mais certeza de que a dívida não vai sair de controle:

> *Quando a dívida alcança o teto, declaramos "alerta de dívida!", as portas são fechadas, todos os novos desenvolvimentos param e todo mundo se foca em limpar o código até que estejam todos de volta à base inicial.*

Idealmente, você resolverá dívidas técnicas durante todo o ciclo de desenvolvimento, mas às vezes você atingirá esse teto, e então será importante parar e trabalhar nele antes que seja tarde demais.

3.4 Juntando tudo

Juntar necessidades de usuário, necessidades de negócio e necessidades técnicas é uma coisa. Descobrir onde colocar toda essa informação e o que fazer com isso é outra coisa totalmente diferente. Você precisa de um sistema para unir e estruturar tudo isso de uma maneira que não apenas faça sentido depois, mas facilite o redirecionamento da informação em diferentes formatos, conforme necessário.

Todo mundo tem suas próprias preferências e seus sistemas para anotar e organizar informação, mas há algumas maneiras de tornar esse processo mais fácil. Primeiro, não use ferramentas independentes — como e-mail ou Microsoft Word — que não se integrem bem com outras. Este tipo de conteúdo em silos é difícil de se compartilhar, de manter atualizado e de formatar para outros propósitos.

Meu jeito preferido de juntar e organizar informações é o formato mais simples de arquivo que temos ao nosso dispor: arquivos de textos simples, preferencialmente taggeados usando um vocabulário controlado. Adoro o fato de que, com texto simples, o foco está nas palavras, e não na formatação. Adoro que é portátil e pode ser

usado em qualquer lugar, em qualquer software que edite ou exiba palavras.

Adoro como é fácil de criar documentos bem formatados quando necessário. Sobretudo, adoro quão rápido é. Eu simplesmente trabalho mais eficientemente desde que mudei para texto simples.

Por exemplo, enquanto usuário de Mac, faço todas as anotações usando o nvALT, de Brett Terpstra. As duas coisas que mais amo no nvALT são:

- Operação não modal na qual buscar notas e criar novas notas acontecem na mesma parte da interface. É altamente eficiente e não há nenhum lag (retardamento).
- Poderosos atalhos no teclado para operar sem mouse, o que acelera sua escrita.

Esses arquivos todos são syncados em uma pasta no Dropbox, o que torna essas notas imediatamente acessíveis em todos meus outros devices (uso Notesy para tomar notas nos meus devices iOS, http://smashed.by/notesy). Mas e quanto à formatação? É aí que entra o Markdown (http://smashed.by/markdown).

O Markdown é uma sintaxe fácil de aprender e discreta, que lhe permite focar no que está escrevendo, sem ficar empacando sobre como ele vai ficar quando você terminar. Ao mesmo tempo, é um sistema poderoso para formatar documentos automaticamente quando você precisa imprimi-los, ou enviar algo para um colega ou cliente. A sintaxe permanece facilmente legível sem atrapalhar no meio de suas palavras.

Quando essas notas têm de ser compartilhadas em documentos mais formais, ainda também é importante ficar longe de ferramentas offline, como Microsoft Word. Os wikis da empresa, ou ferramentas como Google Docs, são ótimos para colaborações e para compartilhar informações sobre as várias pesquisas e necessidades que você coleta com o tempo.

Portanto, considerando que agora você tem uma nova forma de coletar, guardar e compartilhar toda essa informação, a próxima pergunta é o que fazer com isso. Entraremos no assunto de priorização nos capítulos a seguir. Mas, por enquanto, é suficiente dizer que é importante encontrar um equilíbrio entre buscar as necessidades dos usuários, do negócio e técnicas; e o que faz esse equilíbrio mudar se baseia em três principais fatores:

- **O estágio do produto em seu ciclo de desenvolvimento.** É um produto novo ou ele está por aí faz algum tempo?
- **O nível de engajamento do usuário.** Você está tendo dificuldades em ganhar impulso, ou os usuários estão batendo na porta da sua casa para usar seu produto?
- **O estado financeiro do negócio.** Você ainda está tentando descobrir como fazer dinheiro, ou há um fluxo de renda estável?

Dependendo em como esses três fatores são postos juntos, você terá um foco diferente no desenvolvimento do seu produto. O produto é novo, e em uma fase de larga aquisição? Então as necessidades dos usuários devem ter mais peso. O negócio está vendo um crescimento massivo orgânico? Então coloque mais foco em escalabilidade e aumento da renda.

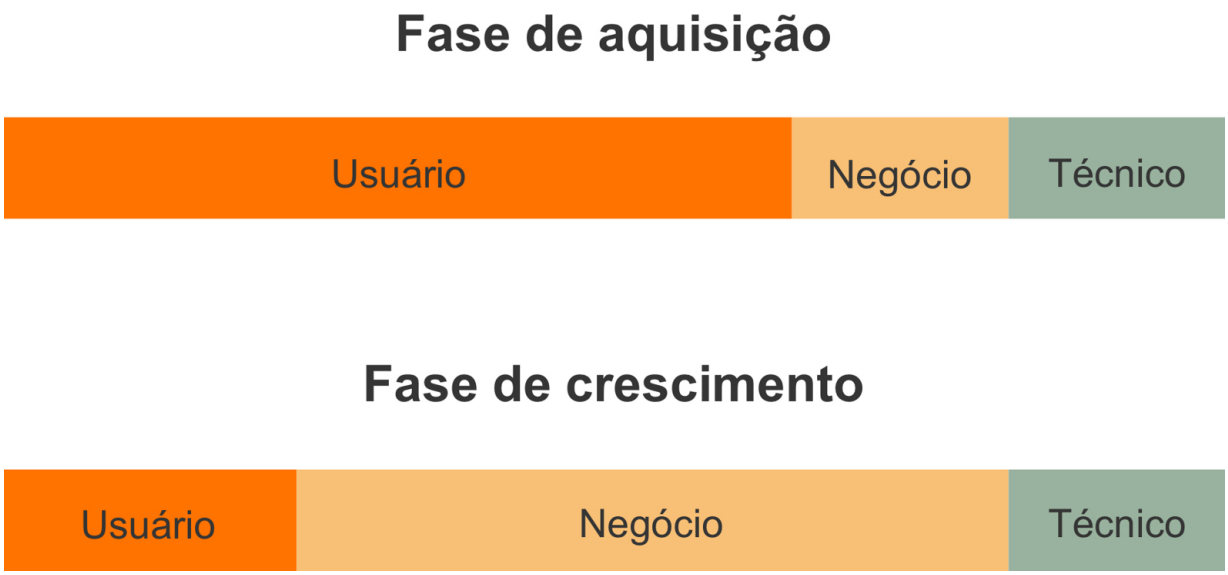


Figura 3.3: Equilibrando necessidades de usuário, de negócio e técnicas, com base na fase do negócio

Essa abordagem vai dar ao gerente de produtos uma ideia bruta de como as prioridades e necessidades deviam ser equilibradas. Entretanto, há um caminho longo entre descobrir o que construir, e como e quando fazê-lo.

O ponto que merece ser salientado é este: sem fazer o trabalho de entender o cerne das necessidades de usuário, de negócio e técnicas, às quais seu produto vai se dirigir, você estará construindo um castelo na areia. O produto pode funcionar por um tempo, mas eventualmente algo melhor vai aparecer. Então, em vez de confiar em hipóteses perigosas, construa um produto sustentável na sólida rocha das percepções reais.

É claro, descobrir as necessidades é uma coisa. Descobrir como transformar esse entendimento em um produto bem-sucedido é algo totalmente diferente. Então, vamos discutir por que tantos produtos falham, e o que podemos fazer para ter certeza de que o nosso não vá por este caminho.

Capítulo 4
Descoberta do produto

Brasília é uma cidade marcante e bizarra. Sendo uma visão do arquiteto Oscar Niemeyer (http://smashed.by/niemeyer), foi construída em apenas quatro anos, de 1956 a 1960. Mais de cinquenta anos depois, sua beleza e elegância são célebres.

Mas a capital do Brasil também é conhecida por algo mais: como é difícil viver lá. Uma "brilhante cidadela" vista de longe, como o Guardian reportou uma vez (*Trouble in utopia as the real Brazil spills into Niemeyer's masterpiece*, http://smashed.by/brazil), de perto Brasília se "degradou a uma expansão violenta, dominada pelo crime e engarrafamentos cacofônicos. O verdadeiro Brasil tem manchado sua visão utópica".

Esse problema ecoa por todo o cenário atual da web também, onde as necessidades de usuários comuns mancham constantemente as visões utópicas das empresas. Por todo o nosso redor, vemos monumentos lindos e vazios, erigidos não para seus usuários, mas para as pessoas que o construíram — e os investidores (*venture capitals*) que os estão prospectando. Mesmo os sites e apps que são mais do que lindos em usabilidade frequentemente falham porque não podem encontrar um mercado grande o suficiente.

Por que alguns produtos interativos não encontram usuários suficientes para ser sustentável? Por que há tantas startups que falharam, apesar de seu novo foco renovado em design? Mais importante, o que podemos fazer a respeito disso?

4.1 A ascensão de produtos usáveis e inúteis

Por muito tempo concordamos que, para um produto ser útil, ele deve ter níveis aceitáveis tanto de utilidade - "se fornece as features de que você precisa" - e usabilidade - "quão fácil de usar e agradável essas features são" (*Usability 1 01 : introduction to usability*, http://smashed.by/usability-101).

Ainda assim, muito frequentemente parece que nós ignoramos o primeiro em favor do último, terminando com várias aplicações fáceis e agradáveis que não têm nenhuma razão para existir. Alguém podia argumentar que a primeira versão do app para iOS Color caiu nesta armadilha (veja em *How color already blew it*, http://smashed.by/color).

Um dos maiores problemas com que novos produtos em particular se deparam é a falta de um product/market fit (http://smashed.by/market-fit), como discutimos no capítulo *1*., onde definimos os papéis e as responsabilidades do gerente de produtos:

> *O product/market fit significa estar em um bom mercado com um produto que o satisfaz.*

O problema acontece quando empresas e startups não passam tempo suficiente aumentando a compatibilidade entre produto e mercado, antes de começarem o design e o desenvolvimento. O conceito da *lean startup* de Produto Mínimo Viável (*Minimum Viable Product*, http://smashed.by/minimum-viable-product) — isto é, apenas aquelas features que permitem o produto ser deployado e nada mais — é certamente útil. Mas em vez disso, devíamos focar em Produtos Mínimos Desejáveis. Veja o artigo de Andrew Chen, *Minimum Desirable Product* (http://smashed.by/mdp):

> *Produto Mínimo Desejável é a experiência mais simples necessária para proporcionar uma experiência de produto de alto valor e satisfatória para os usuários.*

Qual a utilidade de iterações rápidas se tudo o que elas fazem é levar-nos a uma solução abaixo do ideal mais rapidamente, enquanto deve haver uma solução bem melhor por aí? Mas antes de nos precipitarmos e falarmos de como consertar isso, vamos ver algumas das questões de suma importância.

4.2 Por que produtos falham em se encaixar?

O maior problema de Brasília é que os arquitetos que a projetaram não consideraram como a cidade poderia ser usada se milhões de pessoas morassem lá. Eles exibiram uma miopia arquitetural (veja *Shareable: architectural myopia: designing for industry, not people*, http://smashed.by/architectural-myopia), focando-se tão intensamente na tarefa do design, que não conseguiram considerar adequadamente as necessidades das pessoas.

Escrevi sobre um fenômeno similar na nossa indústria: miopia de design (*Designer myopia: how to stop designing for ourselves*, http://smashed.by/designer-myopia). Atraídos pelo reconhecimento (e clientes e investidores) que desejam, os designers frequentemente se focam primariamente em aparecer nas galerias e ser listados em posts de blogs que conduzam toneladas de tráfego.

Não há nada intrinsecamente errado com essa necessidade de reconhecimento, mas isso se torna um problema quando atinge os usuários. Se Brasília nos ensina algo, é que se tornar cego às necessidades dos usuários leva a um caminho perigoso, em que perdemos controle sobre nossos produtos, sem jeito de voltar atrás.

Uma vez que algo foi encaminhado, você pode iterá-lo ou contorná-lo. Iterá-lo é ótimo se você está no caminho certo. Contorná-lo é perigoso porque mudar o curso pode causar prejuízo aos funcionários e aos usuários.

4.3 Descoberta do produto: um jeito melhor

Se queremos criar produtos melhores e mais úteis, precisamos parar de projetar soluções muito cedo e, em vez disso, começar com a descoberta do produto: um processo que nos ajuda a entender o problema devidamente, de modo que não apenas projetemos coisas melhor, mas projetemos coisas melhores.

A descoberta de produto consiste em três passos:

1. Enquadre o problema;
2. Explore e avalie múltiplas soluções;
3. Priorize e planeje.

Vamos discutir cada um destes três passos em detalhe.

Enquadre o problema

É difícil contestar estas palavras, atribuídas a Einstein:

> *"Se eu tivesse uma hora para resolver um problema, eu passaria 55 minutos pensando sobre o problema e 5 minutos pensando sobre soluções".*

O primeiro passo para a descoberta do produto são estes 55 minutos proverbiais. Aqui, o gerente de produtos guia um processo de responder perguntas como:

- Quais problemas e necessidades de usuário estamos tentando resolver? E para produtos existentes, quais as deficiências que precisamos consertar?
- Quais são os perfis dos nossos usuários (personas)?

- Quais insights de consumidores temos disponíveis para informar a solução (suporte ao cliente, analytics, pesquisa de mercado, pesquisa de usuário, análise competitiva e por aí vai)?
- Como o fato de resolver este problema vai ajudar nosso negócio?
- O que torna nosso negócio capaz de resolver este problema?
- Como mediremos o sucesso?

É neste momento que o gerente de produtos pega suas anotações sobre necessidades dos usuários, de negócio e técnicas para implantar e guiar a discussão sobre a essência do produto. Há diversas técnicas para estruturar a discussão e tornar mais fácil chegar ao fundo dessas questões. Diagramas de espinha de peixe (*fishbone*) e os cinco porquês são duas técnicas de análise da causa-raiz que podem ser aplicadas muito eficientemente para definir um problema em termos de necessidades de usuário e objetivos de negócio.

1. **Diagramas de espinha de peixe**

   São usados principalmente como um método de controle de qualidade em indústrias, mas a técnica já traçou um caminho para dentro do mundo do design de produto digital. O diagrama é criado a partir da identificação de um problema ou de um resultado desejado, e então listam-se todos os aspectos que poderiam ser causa do problema ou do resultado.

   Quando utilizado no contexto de desenvolvimento de produto, essas causas geralmente são agrupadas em categorias, comumente referidas como os sete Ps, em inglês: *product* (*service*), *price*, *place*, *promotion*, *people*, *process* e *physical evidence*. Em tradução livre: produto (serviço), preço, posto, promoção, pessoas, processo e evidência física.

1. **Os cinco Porquês**

   Esta é uma técnica similar que usa questionamento iterativo para encontrar a causa raiz de um problema. A ideia básica é declarar um problema e seguir perguntando "por quê?" — às vezes, mais de cinco vezes —, até que um processo em particular possa ser identificado como a principal causa do problema.

Esta fase sempre — sem falhas — produz insights incrivelmente valiosos para o time. As startups ganham clareza quanto ao dizer sim ou não para seu produto. Grandes empresas aprendem como ir além de chavões centrados no cliente, e descobrem quais benefícios eles deveriam estar vendendo a seus usuários.

Como um dos muitos exemplos, uma vez eu estava em um workshop que revelou que os executivos tinham uma visão para a empresa completamente diferente da que os designers e desenvolvedores tinham. Foram duas horas bem estranhas, mas no fim eles concordaram na dura, porém correta, decisão de suspender os planos do seu e-commerce até que algumas áreas de conteúdo no site fossem resolvidas. É ótimo ver o estabelecimento de um propósito emergir dessas sessões — algo que finalmente faça a empresa concordar sobre qual deveria ser o foco do produto.

A partir deste passo, o gerente de produtos produz um diagrama do quadro do problema, o que é simplesmente um resumo visual das principais entregas, na forma de três círculos sobrepostos: necessidades de usuário, objetivos de negócio e competências essenciais.

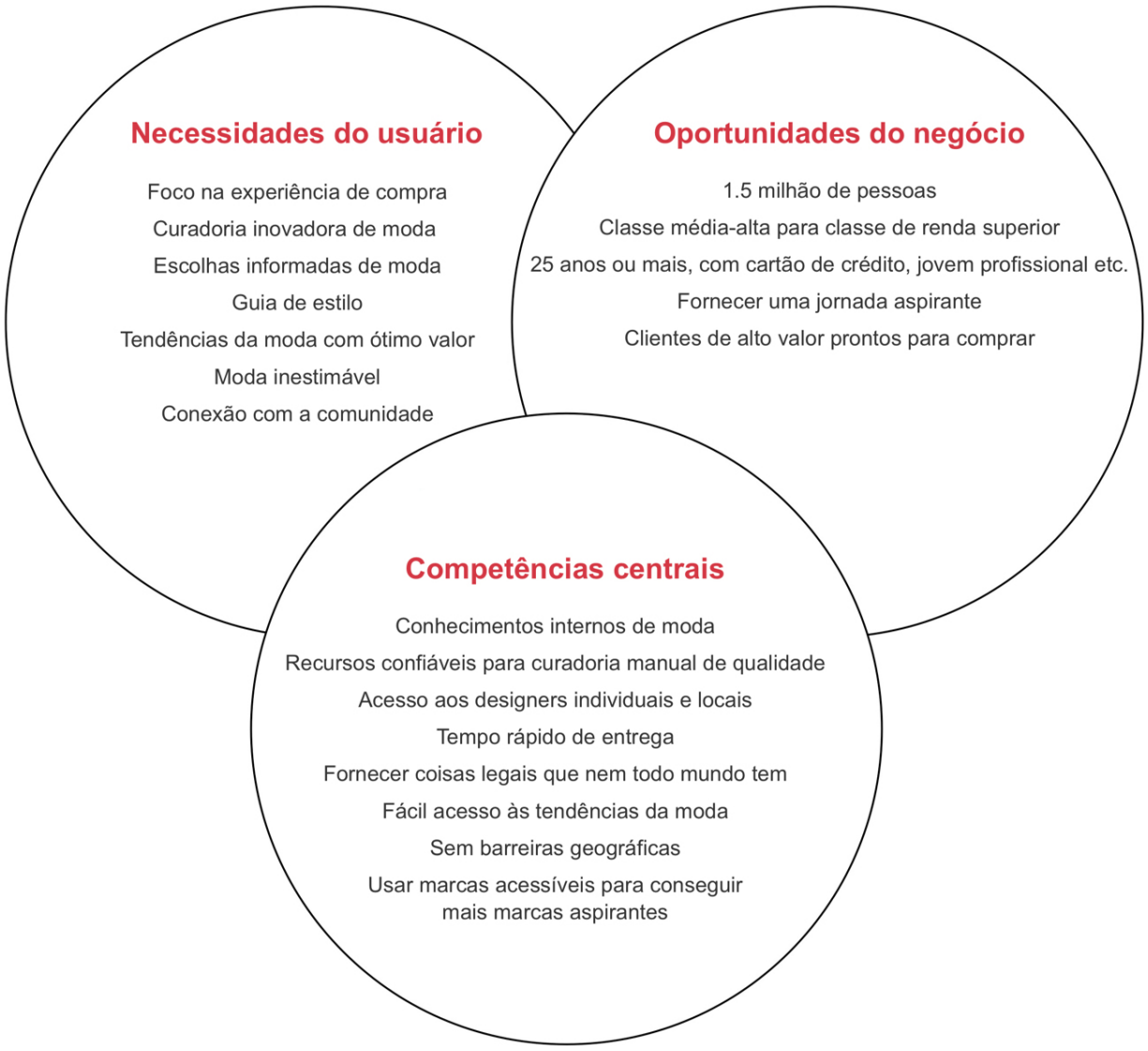

Necessidades do usuário
Foco na experiência de compra
Curadoria inovadora de moda
Escolhas informadas de moda
Guia de estilo
Tendências da moda com ótimo valor
Moda inestimável
Conexão com a comunidade
Oportunidades do negócio
1.5 milhão de pessoas
Classe média-alta para classe de renda superior
25 anos ou mais, com cartão de crédito, jovem profissional etc.
Fornecer uma jornada aspirante
Clientes de alto valor prontos para comprar
Competências centrais
Conhecimentos internos de moda
Recursos confiáveis para curadoria manual de qualidade
Acesso aos designers individuais e locais
Tempo rápido de entrega
Fornecer coisas legais que nem todo mundo tem
Fácil acesso às tendências da moda
Sem barreiras geográficas
Usar marcas acessíveis para conseguir
mais marcas aspirantes

Figura 4.1: Exemplo de um diagrama do quadro do problema (imagem maior em http://smashed.by/problem-frame-diagram)

Cada decisão que o time fizer deve ser ancorada em, pelo menos, um destes círculos — preferencialmente na intersecção entre os três. Decisões de design deviam se forçar em resolver essas necessidades e aproveitar as oportunidades de negócio para capitalizar, usando as competências essenciais identificadas.

Esse também é um bom ponto no processo de criar personas para o seu mercado-alvo. Há ótimos recursos disponíveis para a criação de personas (*The persona lifecycle: keeping people in mind throughout*

product design, de John Pruitt e Tamara Adlin; e *The user is always right: a practical guide to creating and using personas for the Web*, de Steve Mulder e Ziv Yaar), então não entrarei nisso com muitos detalhes além de um rápido resumo.

Personas são personagens arquetípicos e hipotéticos dos usuários, definidas detalhadamente. Todas elas têm nomes e rostos, então o time inteiro consegue imaginá-las. Opostamente a um usuário médio e imaginário, elas são pessoas sólidas, as quais podemos imaginar usando nosso produto para alcançar seus objetivos. Isso é prestativo porque, ao focar em indivíduos que estão mais próximos dos limites da experiência, em vez do mediano, somos capazes de atender a uma porção mais larga da nossa base de usuários.

No documentário *Objectified* (http://www.objectifiedfilm.com/), Dan Formosa, da Smart Design, diz: *"No design, o que precisamos fazer é olhar para os extremos. A parte do meio cuidará de si mesma".* Como exemplo, ele fala sobre como uma vez eles projetaram tesouras de jardinagem especificamente para atender a pessoas com artrite. Eles sabiam que, se a tesoura funcionasse para aquele usuário, ela funcionaria bem para todo mundo. Este é o poder das personas.

Então, desenvolvemos personas por algumas razões importantes:

- **Para captar conhecimento —** O time identifica quais características de usuário importam mais para o design do produto (tais como manjar de tecnologia, tamanho da família e por aí vai), e então usar essas características para escolher usuários que compõem o mercado-alvo.
- **Para criar consenso —** Personas são um bom ponto de partida para os membros do time deixarem claro quem eles acham que são os usuários. Mesmo se todo mundo discorda no começo, é um ótimo processo para chegar a um acordo sobre a quem se dirigir.

- **Para criar empatia —** Constantemente, precisamos ser lembrados de que nós não somos nossos usuários, então nossas necessidades são distantes e secundárias às deles. As personas ajudam a nos colocarmos na mentalidade das pessoas que usarão nossos produtos. É útil pegar todas as especificidades e criar cenários diferentes de como as personas poderiam usar o produto em uma determinada situação.
- **Para guiar um design que maximize a usabilidade —** Personas ajudam a resolver discussões de design. Os times podem estar confiantes de que, se um produto pode preencher a necessidade de cada persona, eles atingiram seus objetivos.

Vale a pena ressaltar que nem todo mundo é fã de personas. As personas podem se tornar caricaturas de usuário simplistas demais, que não levam em conta situações e ações específicas. Sem a devida pesquisa, as personas também tendem a ser superficiais e não muito úteis. Mas esses perigos são fáceis de evitar.

Lembre-se de que personas não são prescritivas, são descritivas. Você não pode identificar uma persona, e então tentar prever o comportamento das pessoas a partir disso. Mas com pesquisa e análise sólidas, você pode usar personas efetivamente para ajudar a focar os esforços de desenvolvimento em usuários-alvo, e ajudar a definir quais features deviam ser incluídas (e, tão importante quanto, excluídas) do produto.

Viajante prático

Willem (52)

Contador
Durbanville

Eu viajo principalmente a trabalho, então me preocupo com segurança e custo acessível

Willem mora em Durbanville, e ele é um contador em um comércio de papel, de tamanho médio. Ele gosta de seu emprego, mas ele realmente vive para os finais de semana, quando ele pode passar tempo com sua esposa e seus 2 filhos, e fazer churrascos.

Willem adora ver esporte na TV, e ele ainda lê Die Burger fielmente toda manhã. Ele dirige um Toyota Corolla, mas seu bem mais valioso é uma moto BMW que ele leva para dar uma volta nos fins de semana.

Willem voa para JHB uma vez por mês, e seu PA aluga um carro para ele por um dia ou dois para andar por aí.

Todo mês de julho, sua família voa para Durban para as férias escolares. Eles ficam com os avós de sua esposa, e eles sempre alugam um carro para poder explorar a área juntos como uma família.

Willem sempre escolhe o carro mais barato e mais seguro para sua família nestas viagens. Ele garante que o carro tenha ar-condicionado, mas escolhe apenas o seguro mais básico.

METAS

- Sair por aí enquanto ele está a trabalho
- Manter sua família confortável e segura durante as férias.

Motoristas de viagem

Figura 4.2: Exemplo de uma persona

Mapas de jornada de usuário são outro resultado útil do processo de descoberta do produto. Mapas de jornada são representações visuais que ajudam a resumir pesquisas, realçar e priorizar necessidades de usuário e oportunidades, e obter a carta branca dos stakeholders. O mapa de um produto é importante porque:

- Confirma o entendimento comum das necessidades de usuários e objetivos, bem como a estratégia que você pretende seguir para atender essas necessidades e objetivos.
- É uma ferramenta excelente de priorização, já que permite empresas a focarem nas partes mais importantes de uma experiência primeiro, sem perder de vista o panorama geral.
- É uma luz de guia para o design. Cada vez que aparece uma ideia de design, uma rápida olhada no mapa da jornada nos ajuda a descobrir se é uma ideia boa e que vai cumprir a estratégia escolhida.
- É um excelente canal para o design que prioriza o conteúdo, o que se encaixa em perfeição com abordagens de design responsivo.

Há vários jeitos diferentes de criar um mapa da jornada de usuário. Ele possui alguns elementos comuns, como uma representação visual do tipo de contato, emoções e os principais aspectos através das suas experiências com um produto. Mas isso apenas é útil até um ponto, então começamos a expandir o conceito.

Somando aos elementos comuns, este documento também se torna uma representação da arquitetura da informação e do planejamento do conteúdo do produto, com personas (necessidades, objetivos e cenários), servindo como um ponto de partida para tudo — o nó que amarra tudo isso junto.

Este documento é um resumo de tudo o que precisamos saber para projetar o melhor produto possível para os usuários. Ele possui os seguintes elementos:

- **Vantagens exclusivas** para nos manter focados no que o site precisa comunicar em todos os momentos. Isso vem direto das necessidades das personas e dos objetivos.
- **Modelos e estágios da jornada** para nos lembrar de como o produto se encaixa na vida das pessoas, e quais chamadas à ação devem estar espalhadas pelo site. Esta seção é uma representação visual da jornada de um consumidor, desde quando descobriram que podiam precisar do produto até bastante tempo depois de começarem a usá-lo. Isso ajuda os gerentes de produtos a manter uma visão holística do produto inteiro e de como ele se encaixa no cotidiano dos usuários.
- **Questões** que nossas personas-alvo provavelmente vão perguntar em cada fase da jornada, para focar o tipo de conteúdo que colocaremos em cada página. Em um contexto de e-commerce, são questões como "Posso confiar neste distribuidor?" ou "Quando meus pedidos vão chegar?".
- **Conclusões das reuniões e princípios chaves** para resumir os pontos de vendas específicos, modelo da jornada e questões do usuário, e documentar como traduzir isso em decisões de design e soluções que precisamos manter em mente através do processo de design.
- **Planejamento de conteúdo** que mapeie cada fase da jornada com as questões que nossas personas vão perguntar durante aquela fase, e o que isso significa para o conteúdo específico que deve ir em cada página. Aqui ficamos bem específicos: nada vai para a página ao menos que esteja no planejamento do conteúdo. E se não conseguirmos identificar uma persona que acharia o conteúdo útil, ele não vai para a lista.

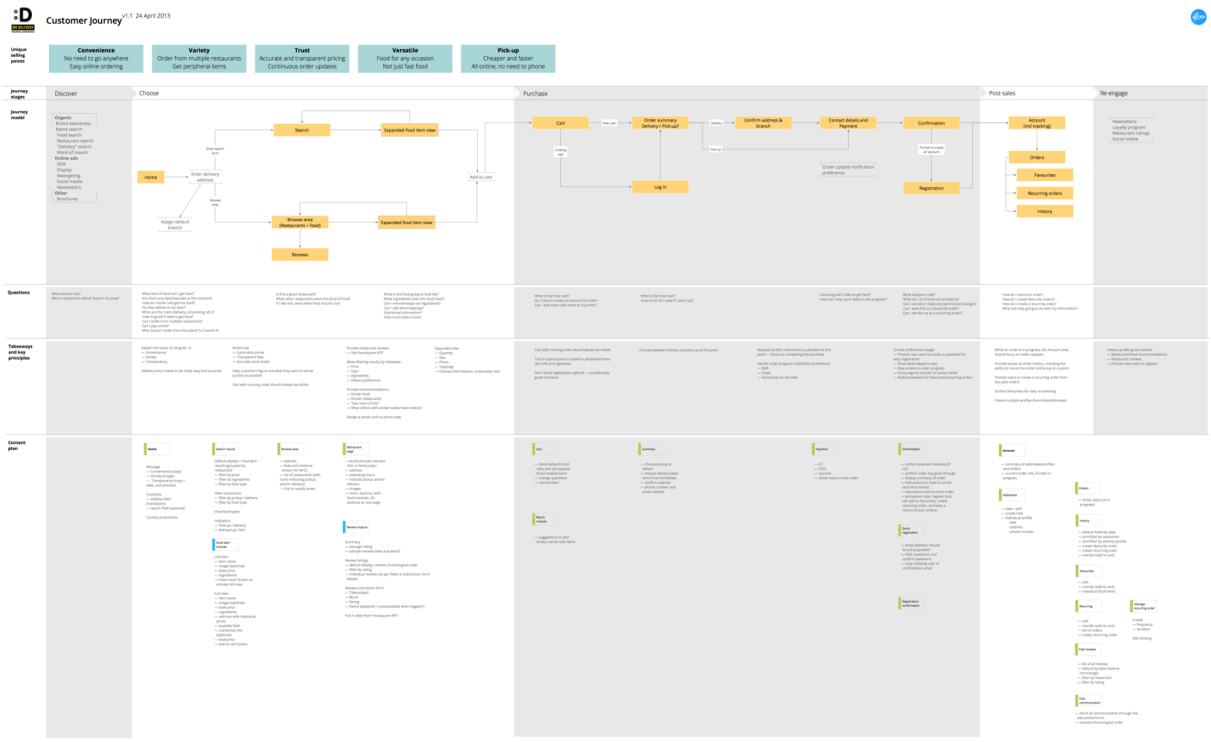


Figura 4.3: Exemplo do mapa da jornada do usuário (tamanho maior em http://smashed.by/customer-journey-map)

Mapas de jornada frequentemente são a culminação de semanas ou meses de trabalho na descoberta do produto. Pode parecer um grande investimento, mas você vai se agradecer uma vez que a fase de execução começar. É um documento que, sobretudo, mantém times sãos porque foca sua atenção no que é importante.

Uma vez que o problema foi definido (e todos os stakeholders concordaram), é hora de começar a pensar nas soluções.

Explore e avalie múltiplas soluções

As conclusões tiradas do enquadramento do produto levam a um período de pensamentos divergentes, no qual você produz tantas soluções diferentes quanto possíveis, o mais rápido possível — visualmente. Abuse dos lápis e de muitas e muitas folhas de papel.

Em vez de abrir seu laptop muito cedo no processo de design, você pode usar esboços para produzir uma variedade de soluções em pouco tempo. É importante gerar tantas ideias quanto possível durante esta etapa do processo. Para entender por que isso é importante, vamos à teoria matemática, especialmente aos conceitos de máximo e mínimo (http://smashed.by/max-min):

> *Na matemática, o máximo e o mínimo [...] de uma função [...] são os valores mais altos e mais baixos que a função assume em um ponto, quer seja dentro de uma determinada vizinhança (local ou extremo relativo) ou no domínio da função, em sua totalidade (global ou extremo absoluto).*

Usando o conceito de vizinhanças locais e funções globais como plano de fundo, vamos olhar a ideia do ponto de máximo local dentro do contexto do desenvolvimento do produto. Pelos propósitos do desenvolvimento do produto, linkei o conceito matemático de vizinhança com ele.

Por exemplo, o iPhone (como um produto) alcançará um ponto de máximo local quando o design atual não pode mais ser melhorado, dados o limite de tempo e as condições do mercado. Ele não é necessariamente o melhor produto que você pode fazer na indústria inteira (máximo global), mas é a melhor iteração do projeto atual.

Para entender isso completamente, temos de diferenciar os conceitos de iteração e variação. Variação é uma maneira de explorar um grupo de soluções alternativas para o produto. Em contraste, a iteração solidifica a ideia que foi escolhida.

Para citar Jon Kolko (*Iteration and variation*, http://smashed.by/iteration-variation): *"Enquanto uma iteração move uma ideia para a frente (ou para trás), uma variação move uma ideia para a esquerda ou para a direita".* Ou, para colocar na linguagem de máximo e mínimo, as variações pesquisam o cenário global para ajudar empresas a escolher a vizinhança correta (produto) para a

qual se dirigir. A iteração, então, os ajuda a encontrar o ponto de máximo local nesta vizinhança escolhida.

É por isso que gerar várias ideias rapidamente por meio de esboços é tão importante. A variação garante que os times passem tempo suficiente para encontrar a vizinhança certa para seu produto, antes de comprarem a terra, por assim dizer, e começarem a desenvolvê-lo. Em resposta para a pergunta *"Eu devo focar em boa experiência de usuário, ou soltar qualquer coisa rapidamente?"*, Ryan Singer escreveu uma boa resposta no Quora (http://smashed.by/ryan-singer), que fornece um resumo de por que variações de soluções de produto são tão importantes:

> *Design é um processo que depende da trajetória. Isso significa que as primeiras decisões condicionam os movimentos posteriores. Na primeiríssima iteração do design, as possibilidades são vastas. O designer define algumas telas e os fluxos de trabalho (_workflows*), e então o programador as desenvolve. Na próxima iteração, as coisas já não são tão vastas. O próximo design tem de se encaixar no design já existente, e o código novo deve se encaixar no código já existente. Código antigo pode ser alterado, mas você não quer ter de se desfazer de tudo. Há uma pressão para seguir em frente com o que já está lá._
>
> *Nossas primeiras decisões de design são como apostas, com cujos resultados teremos de conviver iteração após iteração. Já que este é o caso, há um forte incentivo para termos certeza em relação a nossas primeiras apostas. Em outras palavras, queremos reduzir incertezas nas primeiras iterações.*

Muito tem sido dito sobre o valor dos esboços, mas eu gostaria de acrescentar mais uns pensamentos sobre isso aqui. Embora eu não goste muito da palavra, eu gosto bastante do conceito de *thinkering*, uma palavra que Michael Ondaatje inventou em seu romance *O*

paciente inglês (traduzida para o português como "pensemendando"), para descrever a criação de ideias na cabeça, enquanto fica mexendo as mãos. Michele e Robert Root-Bernsein descrevem como o seguinte (http://smashed.by/thinkering):

> *A manipulação física de coisas, como experiências pessoais diretas de qualquer tipo, gera imagens sensoriais de todas as espécies, assim permitindo o pensamento. Mexer em algo com as mãos (tinkering) leva a mexer em algo com a mente (thinkering). Engajamento corporal com a natureza ensina muito mais do que uma quantidade de palavras ou de números nos livros de ciência. O fazer produz um entendimento pessoal que os símbolos simplesmente não conseguem.*

E é por isso que fazer o esboço é bem melhor para a ideia inicial de um projeto em fase de variação do que pular direto para o design de software. Além do mais, todo mundo pode esboçar. Se você não acredita em mim, o livro de Dan Roam, *The back of the napkin* (http://smashed.by/napkin), vai convencê-lo. É uma ótima fonte para ajudá-lo a descobrir como resolver problemas com esboços.

Os resultados dessa fase do processo são storyboards e esboços simples, de baixa fidelidade, para ajudar a visualizar as soluções possíveis para os problemas que você identificou. O objetivo é colocar essas ideias na frente dos usuários em potencial o mais rápido possível, para pegar o feedback deles e validar suas suposições. Entretanto, primeiro você precisa estreitar essas ideias para chegar a um conjunto maleável; do contrário, você estaria testando hipóteses para sempre e nunca encaminharia nada.

Priorize e planeje

Eu tenho contato com muitos times que reclamam de *analysis paralysis* (paralisia da análise): uma inabilidade de tomar decisões porque há muitos fatores (e pessoas) envolvidas. Bons métodos de

priorização fornecem aos times uma garantia de que, mesmo que eles não estejam focando em todas as coisas de uma vez só, eles estão focados na informação certa para tomar boas decisões.

Você pode fazer isso com uma etapa de pensamento convergente, que restrinja quais ideias e soluções devem ser exploradas mais a fundo. Existem vários processos estabelecidos para este tipo de priorização, cada um projetado para um cenário diferente. Vamos ver três desses métodos.

Método KJ

Com o método KJ (também chamado de Diagramas de afinidade), você agrupa problemas similares, e utiliza um mecanismo de votação para ranquear esses tópicos em ordem de importância. Funciona melhor quando você tem um grupo grande de stakeholders, que terão opiniões fortes sobre o produto, e é preciso tomar decisões rapidamente. Os passos básicos para uma sessão de KJ são os seguintes:

1. Escreva cada ideia ou necessidade em um post-it.
2. Cole todos os post-its na parede, acrescentando mais se ideias adicionais aparecem no processo.
3. Agrupe ideias e necessidades similares.
4. Atribua um nome para cada grupo, mas fique longe de jargões (tente usar uma linguagem clara, focada no usuário).
5. Ranqueie as ideias ou necessidades em ordem de importância por meio de votação. Cada pessoa no grupo tem um certo número de votos (geralmente, três) para dar às ideias ou necessidades que pensam ser mais importantes (use marcadores ou pequenos adesivos como mecanismo de contar os votos). Eles podem distribuir seus votos como quiserem — podem ser os três na mesma ideia, se acharem-na

extremamente importante, ou então distribuí-los em três ideias diferentes.

O método KJ é notavelmente eficiente para se alcançar concordância sobre a prioridade de problemas que precisam ser resolvidos, mesmo em equipes grandes. É um ótimo método que dá voz a cada um, e a chance de defender os argumentos por que eles acreditam que aquela ideia ou necessidade é importante para o negócio.

Modelo Kano

Desenvolvido na década de 1980 pelo professor Noriaki Kano para a indústria automotiva japonesa, o modelo Kano é um método útil para priorizar features de produtos, colocando-as na seguinte escala bidimensional:

- Quão bem uma necessidade de usuário específica está sendo cumprida por uma feature;
- Qual nível de satisfação a feature trará aos usuários.

O modelo é frequentemente usado para classificar features em três grupos:

- **Geradores de ânimo:** features inesperadas, que encantam e tornam o produto tanto útil quanto usável.
- **Compensação de performance:** features que continuam a aumentar a satisfação conforme são feitas melhorias.
- **Expectativas básicas:** features que os usuários consideram óbvias — se elas não estiverem disponíveis no produto, você está com problemas.

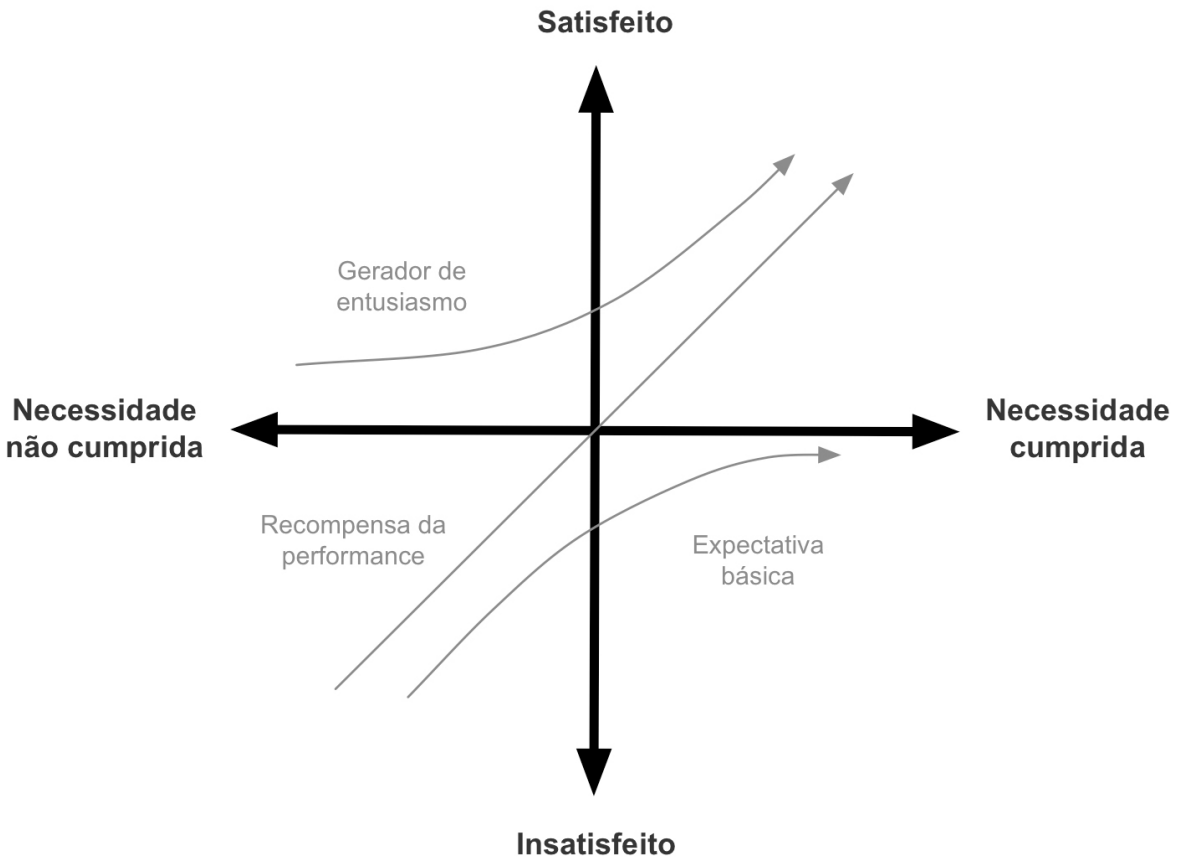


Figura 4.4: Usando o modelo Kano para balancear o desenvolvimento de features

Este método funciona quando você quer garantir que tem um roadmap balanceado, que atenda tanto aos requerimentos básicos quanto às features inovadoras que podem ajudar a alavancar o produto na frente de seus concorrentes.

Abordagem da Amazon.com

A abordagem da Amazon prioriza temas abrangentes primeiro, antes de partir para features individuais e projetos para resolver esses assuntos. Ian McAllister descreve a fase inicial dessa abordagem (a resposta de Ian McAllister para *Product Management: what are the best ways to prioritize a list of product features?*, http://smashed.by/mcallister):

Priorize temas, não projetos. Crie uma lista de temas para seu produto ou negócio. Os exemplos podem ser aquisição de consumidores, ativação, retenção, média de renda por usuário, média de visitas por usuário etc. Escolha aproximadamente três que são os mais importantes para seu produto, dado seu estágio.

Os próximos passos envolvem designar pessoas para trabalhar em cada um desses temas, gerando projetos dentro de cada assunto, e então priorizando cada projeto, com base em seu custo em potencial e no impacto no negócio de cada. É uma boa abordagem quando o número total de features ou de melhorias requeridas é esmagador, e você precisa de uma forma de estruturar e dar sentido a todas elas.

Eu pessoalmente achei que uma versão enxuta do método da Amazon foi mais efetiva e realista nas empresas em que trabalhei. Por mais que eu concorde com os princípios de cada uma das três abordagens que mencionei anteriormente, elas tendem a se tornar pouco realistas no contexto de priorizações em andamento. Então, eis o processo extremamente simples que usamos, com bons resultados.

- Tenha um quadro branco com uma matriz 2x2 permanente nele. O eixo horizontal representa o impacto no negócio (o que inclui necessidades de usuário e considerações técnicas), e o eixo vertical representa o nível de esforço para a implementação (o que inclui pessoas e seus compromissos de tempo).
- Escreva ideias e solicitação de produtos em post-its conforme elas surgem; tenha uma rápida discussão com pessoas relevantes para verificar o valor de negócio e o nível de esforço; e então coloque o post-it na matriz.
- Escreva o número de priorização perto de cada uma das features ou temas, começando com aqueles que possuem o

maior valor de negócio. Eu gosto de uma divisão de 70% e 30% entre features de alto e baixo esforço.

- Há cada uma ou duas semanas, cheque o *roadmap* para ter certeza de que você ainda está trabalhando nas coisas certas, e faça ajustes conforme necessário.

Figura 4.5: Priorização usando indicadores de impacto de negócio e nível de esforço

É um método simples, e está longe de ser perfeito. Mas tem algumas coisas que funcionam bem:

- É detalhado o suficiente para garantir priorização constante com base no que é importante.
- É leve o suficiente para ser prático no uso do dia a dia.

Todos esses métodos funcionam porque facilitam o trabalho em equipe, sem cair nas armadilhas do design por comitê. Cada um tem sua voz, mas nem todo mundo toma decisões. Este é um atributo essencial de todo bom método de priorização, porque, como diz Seth Godin, "Nada acontece quando todo mundo tem de concordar".

Além de fornecer a estrutura necessária para chegar às decisões de prioridade rapidamente, esses métodos também produzem artefatos tangíveis que podem ajudá-lo a vender suas ideias para os stakeholders internos. Experiência de usuário frequentemente é menos um problema de design do que é um problema organizacional.

Por mais que nós simplesmente queiramos fazer nosso trabalho sem obstrução, nós apenas podemos ser realmente eficazes se tivermos um argumento convincente para as pessoas em outras partes da empresa. Esses métodos estruturados de priorização dão este passo razoavelmente indolor, ajudando-o a produzir registros visuais e escritos do seu processo de pensamento.

Mas mais importante do que o método específico que está sendo usado, é que você sai do processo de priorização entendendo a importância e o valor em potencial de cada produto ou feature em que o time pretende trabalhar. Por meio do processo de priorização, gerentes de produtos constantemente restringem as ideias até chegarem a umas poucas selecionadas, que eles desejam desenvolver e testar — e eles precisam estar confiantes de que essas ideias têm as melhores chances de alcançar as necessidades de usuário e objetivos do negócio.

4.4 O resultado

Os artefatos produzidos durante a descoberta do produto dependem do escopo e da natureza do projeto. Às vezes, são apenas alguns

esboços atrás de um guardanapo que um desenvolvedor usa para começar a prototipagem; às vezes, é um grande documento PowerPoint resumindo o processo e os pontos chaves, com o intuito de trazer executivos sêniores para a empreitada.

Independente do real resultado, ao fim do processo, você deve ser capaz de responder as seguintes perguntas com facilidade:

- Qual é o problema que estamos tentando resolver?
- Para quem estamos resolvendo isso? Por que eles deveriam se importar?
- Qual a visão para a solução?
- O que há nisso para nós?
- Qual nosso plano de implementação?

Em algumas empresas, identificar os seguintes itens pode ajudar a montar um plano de produto estratégico que faça transparecer a ideia do product/market fit:

- O produto (qual é a proposição de valor?)
- O mercado (qual o perfil do consumidor?)
- O encaixe (qual o tamanho da oportunidade, o preço e o plano de distribuição?)
- O conjunto inicial de prioridades e métricas de sucesso (isso pode mudar com o tempo).

O real poder do processo de descoberta do produto é que ele vai assegurar seu time de que vocês estão resolvendo os problemas certos para os usuários certos. "Isso tudo é muito bom", vou ouvir você dizer, "mas somos uma startup em rápido desenvolvimento e não temos tempo de ficar sentados conversando". Você tem sim, caso a outra alternativa seja o fracasso causado por um vício

prejudicial por coisas bonitas que levam a 15 minutos de fama, mas nada muito mais do que isso.

Estamos entrando em uma era interessante no web design. Displays de retina podem não ter adoção em massa ainda, mas é apenas uma questão de tempo até que eles virem a norma. Também estamos vendo um nível de interesse em tipografia e gráficos que só experienciamos quando monitores CRT coloridos se tornaram uma coisa.

Há muitos objetos brilhantes por aí, e se focarmos neles (ou em impressionar os investidores, que estão focados neles), negligenciando a utilidade, podemos nos encontrar em uma situação similar àquela de poucos anos atrás, quando construíamos introduções em flash em todo site, só porque era possível.

Em outras palavras, descoberta do produto é essencial para startups, precisamente porque estamos em uma época de tantas inovações visuais empolgantes. Não podemos deixar a fascinação pelo visual nos separar da real utilidade do produto que desenvolvemos.

É verdade que o fracasso nos ensina muito sobre o que funciona e o que não funciona. Mas é tão menos custoso e mais efetivo falhar em várias ideias no papel do que falhar em uma ideia toda desenvolvida e suportada pelos investidores.

Juntos, podemos evitar a construção de Brasílias digitais — projetos que ficam conhecidos, mas não cumprem as necessidades das pessoas que moram lá. Portanto, vamos descobrir antes de construir.

4.5 Estudo de caso: experiência de usuário do kalahari.com

Quando eu comecei a trabalhar no kalahari.com, em dezembro de 2010, o site não fazia melhorias de UI (Interface do Usuário) em mais de dez anos de existência. A descrição do meu trabalho era bem direta: fazer algo a respeito disso.

Eu gostaria de falar sobre o trabalho que nossa equipe fez nos primeiros doze meses para melhorar a experiência do usuário do kalahari.com. Olhando para o site agora, ainda vejo muita coisa errada nele — havia coisas demais que precisávamos consertar. Então esta não é uma tentativa de exibir nosso trabalho como algum tipo de padrão. Faço isso com interesse em compartilhar nossos métodos e as lições que aprendemos nas trincheiras da vida real da gestão de produtos.

Se você entrasse neste site lá em 2010, você provavelmente se sentiria tão sobrecarregado quanto eu. Por onde começar? Em que ordem devemos fazer as coisas? Depois dos primeiros dias, tomando muito café e conversando com pessoas por toda a empresa, percebi que nós tínhamos dois desafios primordiais.

1. **Sem priorização formal ou processo de desenvolvimento do produto**

   Era a mesma situação que eu já tinha visto muitas vezes antes. Os requisitos foram direto do negócio para os desenvolvedores. Isso deu partida a um infinito vai e volta sobre o que era necessário, apenas acenando superficialmente ao design.

   A abordagem *First in, first out* (primeiro a entrar, primeiro a sair) para a priorização também era bem comum. O resultado, bem, não era o ideal. Nós precisávamos consertar isso.

2. **Sem design formal voltado à experiência do usuário**

   Isso não foi nenhuma surpresa, e foi este o motivo pelo qual aceitei o trabalho em primeiro lugar. Não havia pesquisa de usuário, estratégia de conteúdo; não havia design interativo e nenhum design visual além de marketing e materiais de

propaganda. Essa é a parte que realmente me animou: a oportunidade de introduzir o design da experiência de usuário em uma empresa que estava faminta por isso (e isso é um crédito enorme para eles), mas não sabia por onde começar.

Então, nós imediatamente começamos a trabalhar em ambos os problemas.

Olá, eu sou um gerente de produtos

Introduzir uma camada de gestão de produtos em uma empresa que costumava trabalhar sem isso é complicado. Se você o fizer errado, isso pode se tornar um pesadelo político e acabar arruinando suas chances de fazer qualquer coisa boa. Você pode ter as melhores intenções, mas sempre existe o perigo de que a única coisa que as pessoas vão pensar quando olharem para um gerente de produtos é *"Ei, amigo, eu é que costumava ser responsável por essas coisas!".*

Nós certamente não fizemos essa transição perfeitamente, mas acredito que a chave é assegurar-se de conversar com o máximo de pessoas possível sobre quais são seus problemas organizacionais, e como eles acham que eles podem ser melhorados. Você tem de ter um tempo para explicar os benefícios de ter uma equipe para assumir as responsabilidades de estratégia, visão e execução de um produto (e a queda, se ele falhar). E aí, o que é mais importante, você deve tornar o processo de desenvolvimento justo.

Agora, nós tínhamos um time de gerentes de produtos que era responsável por entregar resultados de negócio mensuráveis, por meio de soluções para o produto que encontrassem tanto as necessidades de mercado como os objetivos da empresa. Eles trabalharam bem próximos às suas equipes para desenvolver a estratégia e a visão para seus produtos. Eles asseguraram-se de que os designers e os desenvolvedores fossem incluídos durante todo o processo. E ainda mais importante: eles garantiram que as entregas fossem realizadas.

Olá, eu sou um designer de experiência do usuário

Eu sabia que nós precisávamos construir uma ótima equipe, já que íamos seguir uma abordagem centrada no usuário para identificar e resolver os principais problemas do site kalahari.com. Mas construir times leva tempo e dinheiro, e é difícil justificar a solicitação de uma larga quantidade de funcionários antes de provar que você poderá causar um impacto real para o negócio.

Portanto, nós começamos pequeno. Eu preenchi o papel de UX designer, e nós contratamos um designer visual, já que essa era uma necessidade primordial naquele estágio. Depois, ficamos empacados. Em um site daquele tamanho, e com a pressão que tínhamos para fazer as melhorias rapidamente, optamos por uma abordagem dupla:

1. Fazer algumas mudanças iniciais óbvias no design visual, para melhorar a hierarquia e a estética geral.
2. Ao mesmo tempo, trabalhar em uma estratégia UX de longo prazo, a fim de resolver alguns dos problemas de experiência do usuário no site.

O objetivo era mostrar rapidamente que nós sabíamos o que estávamos fazendo, e então usar esses sucessos para edificar o time mais ainda, e atacar as áreas nas quais poderíamos ter os maiores efeitos nas taxas de conversão.

Construindo um roadmap

Começamos este processo com um time pequeno de três gerentes de produtos e dois designers. Então, inicialmente, não tínhamos o luxo de pesquisas de usuário e de um longo período de descoberta do produto para construir um roadmap. Em vez disso, fomos para a área externa por um dia e construímos um mapa de jornada do consumidor para nossas diferentes jornadas de usuário. Foi um

ótimo jeito de focar no que era o cerne da experiência que precisávamos melhorar.

Além disso, fomos um pouco mais longe. Com base em uma avaliação heurística do site, nós comentamos cada passo na jornada de usuário com as melhorias óbvias que podíamos fazer. Isso nos deu um framework flexível para o ano, e guiou nosso roadmap através dele.

Desde o início, decidimos realinhar, e não reformular. Nossa abordagem era fazer progressos incrementais implacáveis em vez de fazer um projeto de seis a nove meses com um release enorme. Nosso objetivo era lançar uma versão a cada três ou quatro semanas, dependendo do tamanho do projeto.

Em nossas primeiras duas releases, cuidamos de coisas básicas. Essas mudanças tiveram exatamente o efeito desejado. A resposta dos usuários foi imediata e universalmente positiva. Em combinação com algumas boas ofertas, o tráfego começou a aumentar. E, mais importante, fomos capazes de começar a aumentar nosso time e adicionar designers, um pesquisador e um desenvolvedor front-end. O jogo estava começando.

Nós passamos o resto do ano trabalhando sistematicamente por meio do nosso mapa de experiência do consumidor, começando com as áreas mais importantes, nas quais uma experiência de usuário melhor pode trazer os maiores efeitos (registros, checkouts, página de detalhamento do produto e por aí vai). Também pudemos permanecer flexíveis e mudar as prioridades conforme necessário (necessidades de negócio, pressão do concorrente, por exemplo).

Fizemos alguns poucos erros antes de acertar a passada. Às vezes, estimamos o escopo do trabalho incorretamente, tentamos a solução errada primeiro, ou fomos tão pressionados pelo tempo que não conseguimos entregar a qualidade que queríamos. Mas nós lidamos com cada um desses problemas imediata e honestamente.

Admitimos nossas falhas, explicamos nossa frustração com prazos arbitrários, e mostramos como a mudança de prioridades afetava o custo devido à quantidade de tempo que acabava sendo gasta em retrabalho (ou trabalho que nunca é usado). Nós mostramos que seguir uma abordagem de design e desenvolvimento iterativos resultava em um aumento na receita nos nossos fluxos fundamentais. Eventualmente, nosso processo encontrou um bom ritmo, no meio do ano.

- Defina a área em que estamos trabalhando, e defina o que seria o sucesso (quais são as métricas que estamos tentando melhorar?).
- Trabalhe em pequenos times de gerentes de produtos, designers e desenvolvedores para esboçar novos fluxos e desenvolver *wireframes*.
- Teste protótipos com usuários, utilizando o método RITE, de modo que o resultado seja melhores designs, e não apresentações no PowerPoint com recomendações.
- Refine os designs, conforme eles evoluem, para designs visuais de alta fidelidade (com mais testes de usuário, se necessário), e entregue HTML e CSS de alta qualidade como resultado final.

O resultado foi um site drasticamente diferente do que era um ano atrás, com melhorias reais nas métricas de sucesso que estabelecemos para nós mesmos (mudanças positivas nas conversões de registros, checkouts e outros fluxos).

Meu maior arrependimento sobre aquele ano é que não pudemos fazer mais. Fizemos algumas ótimas melhorias para o site, mas ele ainda estava longe de aonde devia chegar. E eu sei que todo mundo do time sentia isso.

Propusemo-nos a construir uma cultura de qualidade acima de tudo, e doeu fisicamente quando tivemos de fazer comprometimentos ou

algo que ia contra esta cultura. Mas certamente foi um ótimo começo.

Capítulo 5
Roadmaps do produto

5.1 A importância dos roadmaps do produto

Nós já cobrimos uma grande parte sobre planejamento do produto e priorização. Também produzimos algumas boas ideias pelo caminho. E agora? Será que nós deveríamos criar um roadmap do produto? Eles saíram de moda nos últimos anos, particularmente em empresas mais voltadas a Agile, como Basecamp, quando Jason Fried proclamou (*Product roadmaps are dangerous*, http://smashed.by/product-roadmaps):

> *Em vez de fazer um roadmap, apenas preocupe-se com algumas semanas de cada vez. Trabalhe na próxima coisa mais importante. De que adianta ter uma lista longa, quando você não pode trabalhar em todas as coisas ao mesmo tempo? Termine o que é importante agora, e então descubra o que é importante a seguir. Um passo de cada vez.*

É difícil discordar de uma pessoa (e de uma empresa) por quem você tem grande admiração, como eu sinto por Jason e o Basecamp. Mas eu acredito, sim, que é importante definir o registro certinho no roadmap do produto — principalmente quando se trata de empresas grandes. O artigo de Jason ressalta duas principais preocupações com roadmaps dos produtos, que resumem a inquietação generalizada que você frequentemente escuta por aí:

- Roadmaps de produto pressupõem que você sabe o que vai acontecer de 6 a 18 meses daqui para a frente.
- Roadmaps do produto estabelecem expectativas, então você não pode mudá-las (e se o fizer, isso se torna um exercício

inútil).

Meu propósito aqui não é discordar do Jason em particular, mas usar o argumento dele para fazer algumas considerações gerais sobre a importância dos roadmaps. Portanto, vamos ver cada um desses pontos separadamente.

Roadmaps do produto pressupõem que você conhece o futuro

Jason escreveu:

> *"Quando você permite ser guiado pelo roadmap do produto, você deixa o passado conduzir o futuro. Você está dizendo: "Seis meses atrás, eu já sabia como as coisas estariam 1 8 meses a partir daqui". Você está dizendo: "Não vou prestar atenção no agora, vou prestar atenção no depois". Você está dizendo: "Eu deveria trabalhar na rede social de amizades psíquicas".*

Não é isso que um roadmap do produto é, nem é isso que ele deve fazer. O propósito de um roadmap do produto é estabelecer uma visão no longo prazo para o negócio, e quebrar isso em peças de trabalho menores e significativas, com base no que você sabe agora.

É uma falácia dizer que se trata de uma lista imutável de datas para onde o negócio é dirigido. Um roadmap do produto que não reage às mudanças do dia a dia do mercado e de dentro da empresa é um documento meio besta.

Nas empresas em que estive responsável pelo roadmap, sempre deixamos muito claro que ele se trata de diretrizes flexíveis que podem (e devem) mudar tão frequentemente quanto necessário. Não é sempre fácil convencer times a ver roadmaps de maneira flexível, mas vale o esforço.

O truque é não descrever o roadmap flexível de um modo que soe como uma desculpa para ser indeciso e não se comprometer com nada. Em vez disso, mostre que um roadmap flexível é a única forma de permanecer proativo quando mudanças importantes acontecem na empresa ou no panorama externo, enquanto se mantém o olhar sobre a visão e as metas do produto.

Roadmaps assim dão aos times e aos gerentes metas objetivas às quais se dirigir. É uma visão comum, um senso de direção que é muito mais do que uma simples conversa fiada — é uma evidência concreta de que estamos nos dirigindo a algum lugar bom e que sabemos como chegar lá.

Podemos mudar a direção quantas vezes quisermos. Isso não o torna um exercício inútil: isso significa que, em vez de começar do zero em um novo roadmap de semanas em semanas, você se ergue sobre seus sucessos passados, não comete os mesmos erros de novo, e continua a fazer progresso mensurável, já que dá para ver o ponto de onde você veio.

Roadmaps do produto estabelecem expectativas erradas

Jason continua mais adiante em seu artigo:

> *"O outro problema com roadmaps é o jogo de expectativas. As pessoas esperam que você entregue o que você diz que vai entregar em 4, 5, 6 meses. Mas e se você tem uma ideia melhor?*
>
> *E se houve uma mudança no mercado que você precisa resolver? E se aquilo que você pensou não foi exatamente o que aconteceu? Qualquer mudança no roadmap invalida o roadmap. Então o mapa absolutamente não é um mapa".*

Se você tem esse problema, não significa que roadmaps do produto sejam algo errado: significa que você está fazendo errado. Enquanto

todas as pessoas na empresa confiarem na natureza fluida do roadmap, você não terá este problema.

Em uma empresa onde eu trabalhei, fizemos isso principalmente por meio de um mecanismo que chamamos de "assembleia do produto" (eu preferia "Força intergalática do produto", mas por algum motivo isso não foi para a frente). Eis como funciona.

A assembleia do produto é composta por investidores de cada departamento da empresa: engenheiros, marketing, suporte, categoria, e por aí vai. Essa sociedade tem uma reunião semanal na qual se discutem o atual roadmap do produto e as prioridades.

Perguntamos a nós mesmos se ainda estamos trabalhando nas coisas mais importantes. Se algo mais importante aparece, ela recebe prioridade mais alta no roadmap, e outra coisa se desloca para baixo; se estamos contentes com a direção, não fazemos nada. Se uma nova oportunidade aparece, perguntamo-nos: "É mais importante do que isto em que estamos trabalhando agora? Ou é algo em que devemos trabalhar a seguir? Se sim, o que vai para baixo na lista de prioridades?".

A partir daí, eu comunicava ao time do produto sobre quaisquer mudanças, e nós discutíamos sobre isso para ter certeza de que nenhuma falha passasse. Porém — e isto é importante —, os gerentes de produto tinham completa autonomia e propriedade sobre a implementação do roadmap. A assembleia do produto estabelece as prioridades (com opiniões de todas as partes da empresa), mas os gerentes de produto trabalham com seus times de desenvolvimento (e outros) para estabelecer o cronograma, os detalhes de implementação, o design, tudo.

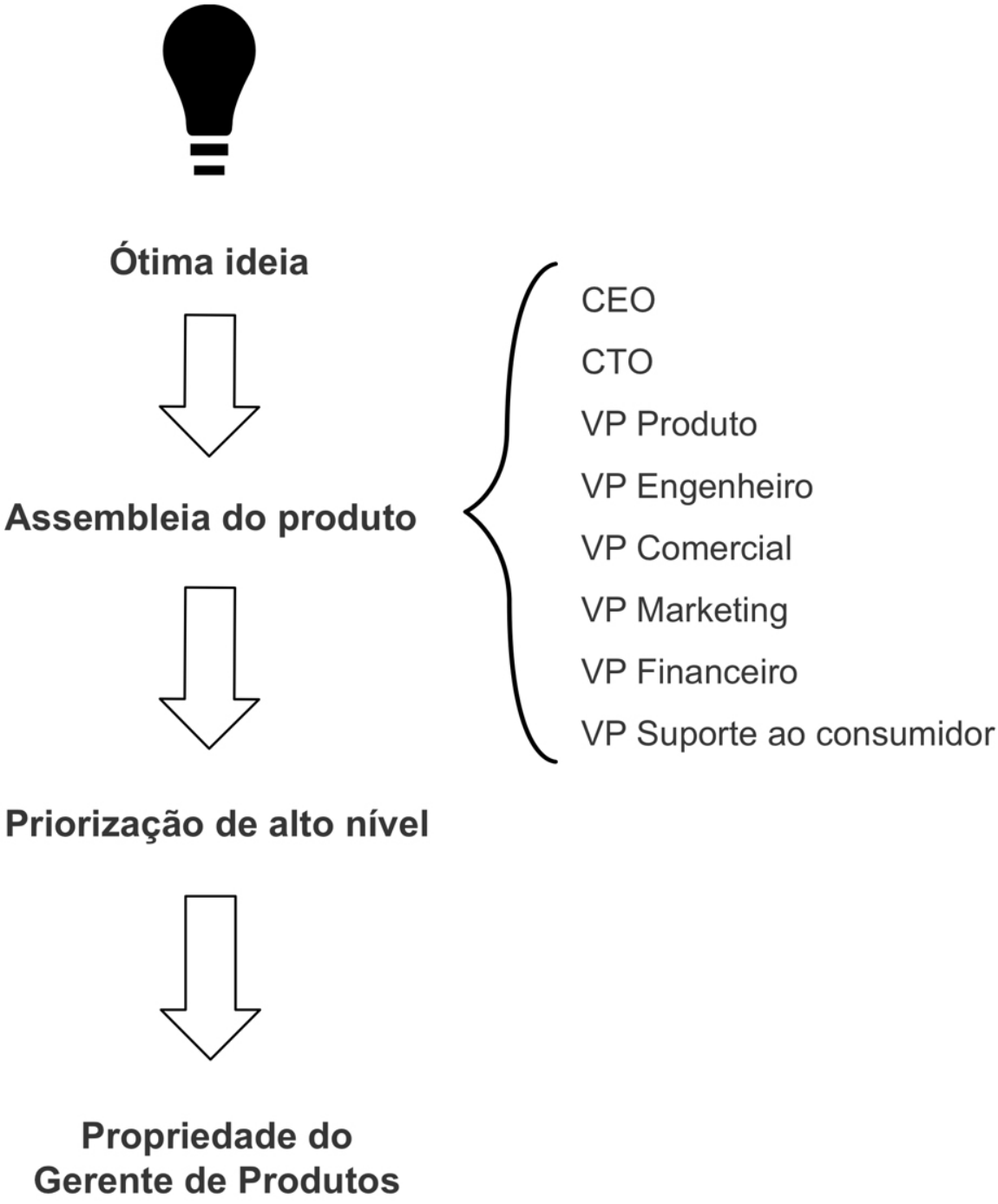


Figura 5.1: Como a assembleia do produto funciona

Esse processo tem três principais vantagens. Primeiro, ele dá ao time de gestão uma completa transparência sobre aquilo em que o

time do produto tem trabalhado, e permite que qualquer um possa defender o caso de uma mudança nas prioridades. Essa transparência dissolve a vasta maioria da politicagem que você vê em muitas empresas, e liberta os times para fazer o que fazem de melhor: executar.

Segundo, isso previne *scope creep*, isto é, o aumento incontrolável do escopo do projeto devido a mudanças de decisões. Nada pode entrar no roadmap sem que outra coisa vá para fora ou para baixo. Como qualquer pessoa que tenha trabalhado em uma grande empresa sabe, esta é uma parte crítica de um ciclo de desenvolvimento bem-sucedido.

Finalmente, ele dá ao gerente de produtos e aos seus times o que eles precisam para ter sucesso: direção e autonomia. Como Jocelyn Glei disse: *"Dê aos membros do seu time o que eles precisam para florescer, e então saia do caminho deles"* (*What Motivates Us To Do Great Work?*, http://smashed.by/motivate).

Por que roadmaps do produto são seguros (e essenciais)?

Em um nível prático, fiz o exercício de imaginar como poderíamos operar em uma empresa sem um roadmap. E, apesar de isso poder até funcionar em algumas circunstâncias, em geral me parece uma proposta muito perigosa. Mudanças nas páginas e nos fluxos atuais afetam mudanças que faremos posteriormente — o gerente de produtos deve pensar sobre isso.

Se você encarar seriamente a mudança incremental frequente em oposição a um grande redesign dos projetos, você não pode viver sem um roadmap, porque você não terá ideia de quão longe você teria ido, ou o que você ainda precisa fazer, e se algo é mais importante do que outra coisa. E talvez o mais perigoso de tudo: todo mundo na empresa virá até você e exigirá que todos os seus projetos sejam feitos imediatamente, e você não terá um método sistemático para lidar com isso de uma forma que seja o melhor para os negócios. Andy Wagner resumiu meus sentimentos em seu

artigo, bem sucintamente, em um comentário no post do Jason no 37signals (http://smashed.by/roadmaps-comment):

> *"Roadmaps do produto são uma oportunidade de sonhar sobre o que o futuro pode parecer, de modo que, assim como você realiza suas respostas ao cliente no dia a dia, você pode fazê-lo consistentemente com a construção do estado futuro. Enfaticamente, isso não deve ser algo a que se escravizar; deve ser algo dinâmico e especulativo em vez de estático e específico".*

Jason escreveu: *"Quanto mais fundo você vai a partir de agora, menos você sabe. E quanto menos você sabe, pior suas decisões serão".* Nós concordamos nisso. Meu argumento é que, sem um roadmap, você apenas vê o agora. E se você apenas vê o agora, sem ver o ontem e o amanhã, você não vê o todo. E "quanto menos você sabe, pior suas decisões serão".

5.2 Os elementos de um roadmap

Então, como é um roadmap do produto? Mais uma vez, isso depende completamente do que é confortável para a empresa. Minha forma favorita de comunicar um roadmap é o mais leve possível — quadros brancos pelo escritório ou wikis da empresa. É perigoso introduzir documentos e sistemas internos com os quais as pessoas não estão acostumadas. Então, no que for que a pessoa já estiver trabalhando, use esta plataforma. Muitas empresas já possuem um wiki interno, portanto esse é um bom lugar para se começar.

Também é muito importante tornar o roadmap o mais visível possível para o resto da empresa. Discutiremos mais sobre isso quando falarmos de especificações funcionais, mas gerentes de

produto trabalham abertamente. E eles fazem isso porque seus objetivos não são a glória pessoal; o objetivo é um ótimo produto.

Para isso acontecer, eles não podem se apegar às suas próprias ideias. Também não podem correr o risco de fazer coisas em segredo que possam afetar o produto ou sair pela culatra politicamente.

Na minha visão, roadmaps do produto devem conter os seguintes elementos:

- A prioridade do projeto (1 sendo a mais alta, e por aí vai).
- Uma descrição detalhada com não mais do que cinquenta palavras.
- Uma lista de contatos envolvidos no projeto (gerentes de produtos, negócios, líder de desenvolvimento).
- Uma indicação do impacto do negócio e o nível estimado de esforço.
- Um link para a especificação funcional assim que disponível.

Você vai reparar que há algo visivelmente faltando nessa lista: datas. Pode ser controvérsia, mas eu acredito que roadmaps do produto não devem ter datas associadas aos projetos, apenas prioridades.

É aí que minhas visões começam a convergir com o princípio geral de Jason em seu post — trabalhe na coisa mais importante até concluí-la, e aí siga para a próxima, e a próxima. Eis um exemplo de como seria um roadmap flexível, incluindo esses elementos mencionados:

High Level Product Priorities - 2010 Q4

- Legend
- Buyer
- Search and Backoffice
- eContent, Mobile, APIs
- Seller
- Customer journey

Legend

- In progress
- Problems - attention required
- Completed

Buyer

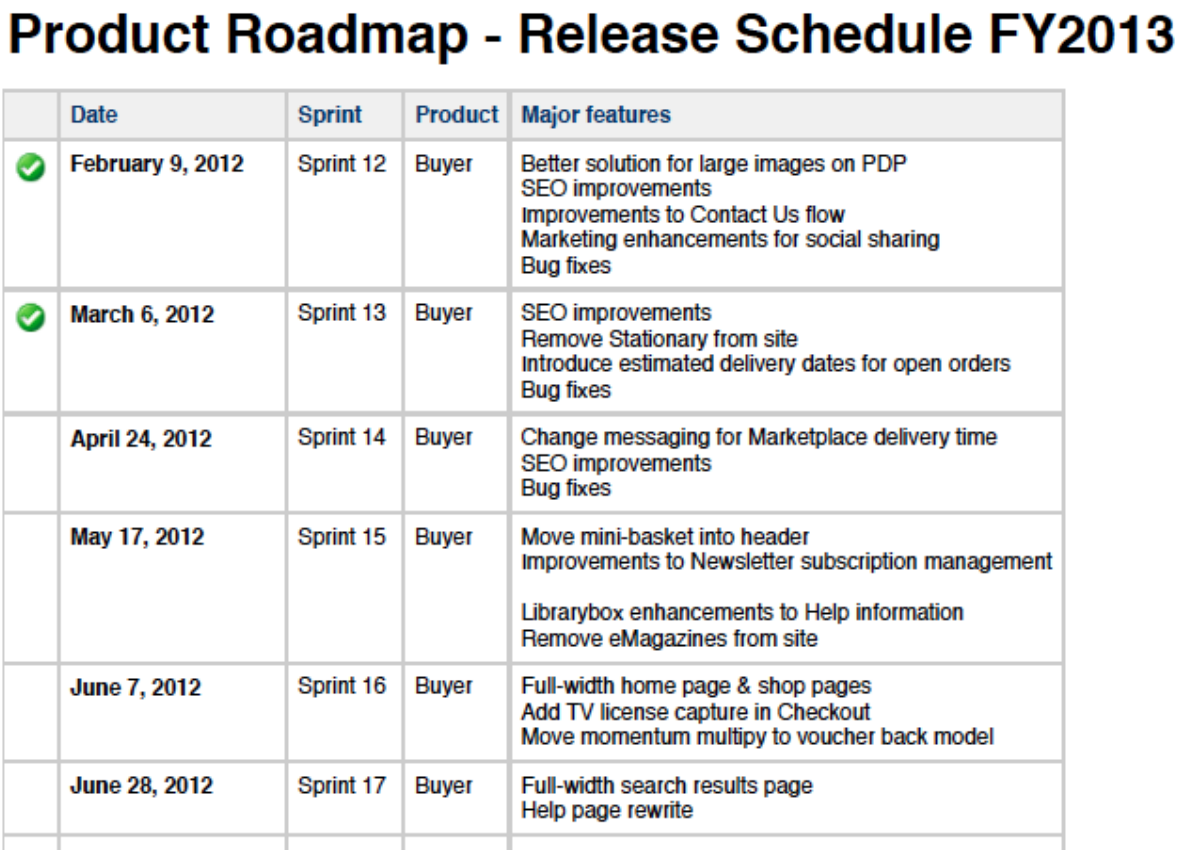
Priority	Status	Summary	Detail	Business	PM	Dev lead	Business impact	Effort	Spec
-		Visual refresh of header/footer	Visual changes to the header and footer, as well as changes to reduce rounded corners and replace buttons	Rian	Rian	Anthony/Dylan	Low	Low	Wiki
1		Credit card immediate settlements	Immediate settlement for credit cards + automatic on hold for foreign orders + automated fraud checker in real time	Gary	Hein	Garth	High	High	BUYEF
2		Replatforming interface	Frontend and platform work for replatforming (Framework, Header/Footer, Home, Product pages, SRP, etc.)	Mike N	Hein/Rian		High	High	
2		Visual refresh phase 2	Global font change, one-colour banners, left nav changes, replace old promo block	Rian	Hein	Lorinda	Medium	Medium	BUYEF
2		Checkout redesign	Redesign of checkout pages, limited to the pages themselves (no combining of pages or flow changes)	Rian	Hein		High	High	BUYEF
2		Login/registration redesign	Redesign of login/registration page	Rian	Hein		Medium	Medium	BUYEF
3		New eTrader platform	Redesign of affiliates site, including required intefaces with Hybris/Endeca/SAP	Liz					
3		Profile page redesign	Redesign of Profile page, including rewrite of copy	Rian	Hein		Medium	Medium	

Figura 5.2: Exemplo de um roadmap flexível com prioridades, mas sem datas (tamanho maior em http://smashed.by/roadmap)

É claro, é impossível se livrar completamente das datas, então eu prefiro fazer isso no cronograma de releases. Este cronograma lista as datas em que a empresa planeja lançar código (frequentemente no intervalo de dois ou três semanas, se a empresa usa

metodologias ágeis), e indica quais projetos estão programados para ir ao ar em qual data.

As datas dos lançamentos de código podem não mudar, mas quais projetos e features entram em qual release podem. Se algo der errado, ou muitas quinas devem ser aparadas, um projeto pode passar para a data do próximo release sem mudar sua prioridade, ou então comprometer a qualidade apenas para alcançar uma data arbitrária.



Product Roadmap - Release Schedule FY2013

	Date	Sprint	Product	Major features
✓	**February 9, 2012**	Sprint 12	Buyer	Better solution for large images on PDP SEO improvements Improvements to Contact Us flow Marketing enhancements for social sharing Bug fixes
✓	**March 6, 2012**	Sprint 13	Buyer	SEO improvements Remove Stationary from site Introduce estimated delivery dates for open orders Bug fixes
	April 24, 2012	Sprint 14	Buyer	Change messaging for Marketplace delivery time SEO improvements Bug fixes
	May 17, 2012	Sprint 15	Buyer	Move mini-basket into header Improvements to Newsletter subscription management Librarybox enhancements to Help information Remove eMagazines from site
	June 7, 2012	Sprint 16	Buyer	Full-width home page & shop pages Add TV license capture in Checkout Move momentum multipy to voucher back model
	June 28, 2012	Sprint 17	Buyer	Full-width search results page Help page rewrite

Figura 5.3: Exemplo de cronograma de releases com datas e features dentro da release (veja em tamanho maior em http://smashed.by/release)

5.3 A seguir...

Com seu planejamento estratégico de produto em seu devido lugar, bem como um roadmap e um cronograma provisório de release para guiar seus passos, é hora de começar a executar e encaminhar as coisas.

Discutimos a fase de planejamento do produto da gestão de produtos isoladamente, mas é claro que não é assim que funciona na prática. Tudo o que foi mencionado na Parte 1 deste livro acontece continuamente conforme o produto evolui.

Mas, por mais importante que seja o planejamento, você não pode fazer isso para sempre — qual a graça disso? E é aí que entra a Parte 3. Vamos falar sobre a execução do produto — o processo de validar suas ideias com os consumidores, e construí-las do modo mais eficaz.

Parte 3 — Execução

Capítulo 6
Definindo um produto

Na primeira metade deste livro, passamos bastante tempo discutindo por que gestão de produtos é importante, com o que se parecem os gerentes de produtos, e como ter certeza de que você está usando seu tempo para construir os produtos certos para as pessoas certas.

Foi um tempo bem gasto, já que, conforme falamos longamente, é bem mais custoso construir o produto errado do que descobrir mais cedo no processo de design e desenvolvimento que você não está realmente alcançando o ponto. Agora que discutimos sobre como funciona um planejamento contínuo de produto e como criar um planejamento de produto estratégico, é hora de focar no processo de execução.

Nesta seção, discutiremos as várias atividades necessárias para dar vida ao produto, e como trabalhar com diferentes times para alcançar seus objetivos. Também discutiremos brevemente o processo de design centrado no usuário, que é um componente essencial ao longo do processo de execução, para assegurar que a empresa mantenha sua direção voltada ao mercado.

Gerentes de produtos devem se assegurar de que as equipes de experiência do usuário (UX) estejam sempre envolvidas no processo. Mesmo se um gerente não realizar por conta própria todas as atividades do design centrado em usuários, é importante entender o processo e usar os resultados na tomada de decisões.

A seguir, está um diagrama que mostra o resultado desta seção.

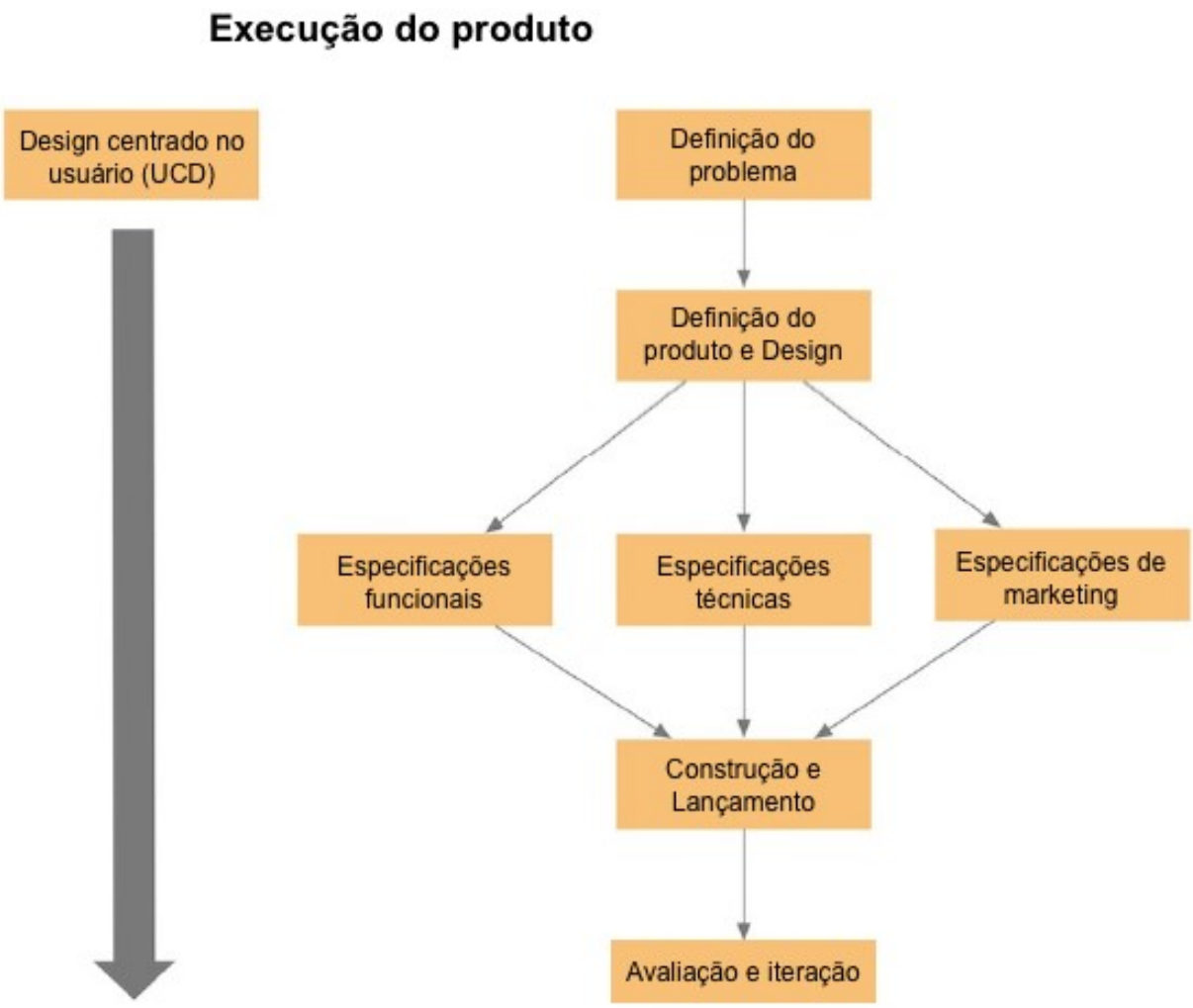


Figura 6.1: O processo de execução do produto

Vamos começar com a discussão sobre a definição de problemas e produtos, e como elas são diferentes de especificações funcionais e técnicas. Depois, faremos uma digressão para falar sobre design centrado no usuário e a importância de teste de protótipo com usuários-alvo. Então aplicaremos esses princípios para o resto do processo — construção e lançamento de features, e a avaliação dos seus sucessos.

6.1 Definição do problema

Requerimentos frequentemente ganham uma má reputação, especialmente no contexto de princípios lean e desenvolvimento ágil, em que se pressupõe que definir requerimentos se trata de descobrir quais features os *stakeholders* acham que deveriam ter em um produto antes de você começar a projetar e a construí-lo. Pode ser assim que os requerimentos eram vistos nos tradicionais processos cascata, mas não é uma boa forma de construir produtos.

Eu sempre imaginei um gerente de produtos com um jaleco e uma prancheta, andando pelo escritório perguntando a pessoas aleatórias quais features eles gostariam que um produto tivesse. Pode parecer como colaboração, mas essa abordagem tem a ver principalmente com politicagem - fazer com que todo mundo sinta que tiveram uma participação. Colaboração é essencial, mas nós já falamos sobre as melhores maneiras de erguer a colaboração no processo.

O tradicional agrupamento de requerimentos é focado demais em features, e raramente leva em conta as necessidades do usuário. Então, em vez disso, precisamos definir requerimentos de um modo um pouco diferente. Com isso em mente, vou me referir a esta parte do processo como a fase de definição do problema.

É importante diferenciar entre a definição e a especificação de um problema. A definição é uma curta declaração do problema que você está tentando resolver. Uma especificação explica como resolvê-lo.

Se a seção de planejamento do produto nos ensinou algo, é a importância de manter o problema que você está resolvendo na frente e no centro o tempo todo. É por isso que é tão importante quebrar essas ideias em definições do problema primeiro, antes de começar a trabalhar em soluções (ou hipóteses, no contexto de metodologias lean), de modo que você não perca de vista o problema que está sendo resolvido.

Uma boa maneira de escrever as definições do problema é usar um formato similar ao das histórias de usuário, no desenvolvimento ágil. Eu as chamo de *histórias do problema*: **[usuário]** ***tem*** **[problema]** ***quando*** **[gatilho]**.

Por exemplo, em um produto de serviços de finança, um gerente de produtos pode ter uma definição de problema que afirma que: "Investidores não são capazes de submeter documentos online de suporte quando eles precisam fazer mudanças nos portfólios dos clientes." Isso se torna uma declaração do problema que precisa ser resolvido por meio de melhorias no produto. A adição do elemento de gatilho também assegura que você permaneça focado na causa da ação que os usuários precisam tomar para achar o produto útil.

Uma definição eficaz do problema inclui os seguintes elementos:

- Ela define claramente os usuários-alvo (que também podem ser o próprio negócio ou o time de desenvolvimento).
- Define claramente o problema que precisa ser resolvido.
- Define claramente em quais circunstâncias o problema ocorre (os eventos gatilhos).
- Possui documentação de suporte o suficiente para fornecer contexto sobre os usuários-alvo e o problema para guiar o projeto da solução.

Para reiterar, essa fase não se trata de fazer uma lista do que todo mundo no escritório quer ver em um novo produto. Mas infelizmente é isso o que frequentemente acontece — todo mundo entra em uma sala, e divaga sobre suas listas de desejo de quais features deviam estar em um produto, com base em seus próprios sentimentos e preferências. Aí, o pobre gerente de produtos tem de classificar tudo isso e criar um produto coerente.

Não. Vamos parar com este comportamento maluco. Em vez disso, vamos trabalhar em fornecer uma definição clara de um problema

tratável para resolver para um grupo específico de usuários, e usar nosso tempo criando as soluções certas.

De onde vêm essas definições do problema? Bem, a resposta devia ser bem fácil nesse ponto: suas definições do problema são tiradas da lista de necessidades priorizadas (do usuário, negócio e técnicas), conforme dispostas no planejamento estratégico do produto.

Por exemplo, se o próximo projeto com prioridade é um redesign para a conta ou perfil de um e-commerce, o gerente de produtos pegaria todo o background da fase de planejamento do produto, e destilaria isso até uma simples história do problema, como esta:

> *Compradores não conseguem encontrar a informação que procuram quando querem verificar o status de seus pedidos.*

Essa história do problema forma a base do projeto que mantém todo mundo fixo no usuário, no problema e no contexto do produto.

6.2 Definição do produto e design studio

Uma vez que você selecionou uma definição do problema para trabalhar em um projeto específico, é hora de pensar em possíveis soluções para os problemas identificados. Aqui o time usa o tempo criando ideias de produto. É similar ao que o movimento lean se refere como formulação de hipóteses: a elaboração de suposições ou ideias com o objetivo de testá-las por meio da pesquisa de usuário.

No capítulo *3. Descoberta do produto*, sobre a descoberta do produto, falei sobre a importância da variação antes que um processo iterativo comece. Se o time pensou e rascunhou bastante durante a descoberta, use estas ideias como um ponto de partida

para formalizar soluções possíveis na forma de protótipos que possam ser testados e validados com usuários potenciais, antes que elas sejam construídas.

Eu usei principalmente o Axure RP (http://www.axure.com/) para criar protótipos interativos, mas existem várias outras ferramentas e frameworks disponíveis para prototipagem, desde soluções específicas de código (a ZURB Foundation, em http://foundation.zurb.com/, e o Bootstrap, em http://getbootstrap.com) até para unir protótipos de papel (POP for iPhone, em https://popapp.in/).

O tipo de protótipo não importa, contanto que seja realístico o suficiente para fornecer um feedback real de usuário. Em geral, wireframes estáticos não são tão úteis, já que eles não permitem interação com o usuário, e também não são tão prestativos como uma forma para os desenvolvedores descobrirem o que o produto faz (a menos que você passe bastante tempo comentando os wireframes, o que não é um jeito de usar seu tempo).

Frequentemente, não é factível sair por aí testando um monte de protótipos de soluções para resolver o problema. Nesses casos, é útil criar diversas variações como um grupo, e depois refiná-las a uma ou duas soluções para serem validadas em testes de usabilidade.

Uma forma eficaz para fazer isso é realizar um *design studio* com o time do projeto. Um workshop de design studio envolve o time inteiro no processo para desenvolver e refinar ideias em um curto período de tempo — frequentemente, em algumas horas. Um design studio deve incluir membros de times de toda a empresa: designers, estrategistas de conteúdo, desenvolvedores, gerentes de produtos, marketing e por aí vai. Há muitas maneiras diferentes de gerir um workshop de design studio, mas geralmente há alguns elementos comuns.

Comece com uma discussão sobre a definição do problema na qual você está trabalhando, de modo que todo mundo tenha um entendimento comum sobre a área em questão e as personas cujos problemas você pretende resolver. Então, continue com três (ou mais) iterações de esboços, usando o mesmo processo com cada ciclo: rascunhar, apresentar, criticar, refinar.

- Iteração 1: rascunhem individualmente, usando um template de 6 variações por página.
- Apresente para o time, receba críticas.
- Iteração 2: escolha a melhor ideia, esboce um conceito refinado individualmente, usando um template de apenas uma variação por página.
- Apresente para o time, receba críticas.
- Iteração 3: escolha as duas ou três melhores ideias, esboce um conceito refinado como um time, usando pedaços grandes de papel.

Já que praticamente ocorre bastante crítica durante o design studio, é importante ter certeza de que o grupo sabe como dar bons feedbacks. Discuti sobre boa crítica em detalhes na Parte 1, então não vou me repetir aqui, exceto realçar os componentes importantes que deveriam ser reiterados durante cada sessão de design studio:

- Deixe cada pessoa explicar inteiramente seu esboço ou ideia.
- O time deve primeiro apontar o que eles gostam naquele esboço, de forma que o apresentador saiba em quais direções seguir.
- Exprima as críticas como questões, para dar ao apresentador a chance de responder e explicar seus pensamentos por trás de suas decisões.

Ao fim deste processo, você terá um pequeno número de ideias que estão prontas para prototipagem e testes com usuários. Você também estará seguro de que testou muitas diferentes variações em um curto período de tempo, e refinou aquelas ideias em algo que você acredita que vá resolver o problema da melhor maneira possível.

Agora é hora de envolver seus usuários alvo. Vamos falar sobre isso dentro do contexto do design centrado no usuário.

Capítulo 7
Design centrado no usuário e workflows

Testes de usabilidade têm sempre sido um pilar do meu trabalho de produto. Minha jornada para gerente de produtos começou no time de pesquisa de experiência do usuário no eBay, logo eu olho para a maioria dos trabalhos de produto pelas lentes do design centrado no usuário (UCD — *User-Centered Design*).

Conforme mudei para uma função de agência mais tarde na minha carreira, as coisas ficaram um pouco complicadas. Acabou que teste de usabilidade é algo difícil de se vender para clientes. Sempre aparecem as mesmas preocupações: vai demorar muito, dez testes não são o suficiente, e por aí vai. Mas eu aprendi a continuar pressionando, e manter-me firme na decisão de não pegar um projeto se ele não incluísse teste de usabilidade.

Logo percebi algo muito importante. A briga dos clientes contra testes de usabilidade dura apenas enquanto eles não observarem o primeiro teste. Aí vem uma luz e eles começam a nos dar dinheiro para isso (ok, não é verdade, mas isso seria legal). Basta ver um usuário tendo dificuldades com seu produto para ver o valor desses testes.

Considerando o quão essencial é este tópico, gostaria de fazer uma pequena digressão neste capítulo para falar de design centrado no usuário com mais detalhes. Esse não é um passo separado do processo de desenvolvimento do produto. Pelo contrário, é uma metodologia que devia acompanhar e guiar o processo do começo ao fim.

Faz sentido se dirigir a isso neste ponto porque, conforme você entra na fase de execução da construção do produto, UCD se torna cada vez mais importante para assegurar que você não saia

construindo algo cegamente no escuro, mas teste suas ideias com usuários reais do mercado-alvo.

O valor do UX design tem sido discutido com detalhes em vários lugares, então vou assumir que você tem um entendimento básico do valor de um bom design, e que todos nós queremos uma boa experiência nos nossos produtos. Em resumo, queremos criar produtos que são úteis, usáveis e deslumbrantes. E queremos integrar isso com o que é tecnicamente possível, lucrativo, e se encaixa nos nossos objetivos estratégicos. É isso que o UCD busca alcançar.

Há um benefício de UCD em particular em que gostaria de focar aqui: a metodologia ajuda as empresas a gastar menos tempo construindo coisas erradas. Seguir um processo de design centrado no usuário reduz drasticamente os custos do desenvolvimento e manutenção. Karat (1993) aponta que:

> *Oitenta porcento do custo do ciclo de vida do software ocorre após o produto ser lançado, na fase de manutenção. Desse trabalho, 80% é devido a requerimentos de usuário não atendidos ou não vistos, e apenas 20% é decorrente de bugs ou problemas de confiabilidade.*

Albert Lederer e Jayesh Prasad descobriram o seguinte (1992):

> *63% dos projetos de software excedem suas estimativas de orçamento por conta de quatro principais razões, sendo todas relacionadas à usabilidade do produto: pedidos frequentes por mudanças feitos pelos usuários, tarefas negligenciadas, falta de entendimento que os usuários têm de seus próprios requerimentos, e comunicação e entendimento insuficientes sobre a análise de usuários.*

No livro *Cost-justifying usability* (BIAS; MAYHEW, 1994), os autores apontam que se o custo de realizar mudanças de design durante a fase de design centrado no usuário é tomado como base, as mesmas mudanças custariam dez vezes mais durante a fase de desenvolvimento, e cem vezes mais depois que o produto é lançado.

Portanto, o quanto antes você descobrir necessidades e problemas, mais dinheiro você economiza no desenvolvimento. Faz sentido, e mesmo assim é mais difícil convencer empresas (e às vezes até nós mesmos) a fazerem as coisas certas na primeira vez, do que é convencê-los a fazer isso depois que algo falhar.

Isso não quer dizer que o produto tem de ser perfeito e completo desde o primeiro dia. Mas significa que, em vez de lidar com uma falha de ignição em termos de necessidades de usuário, você terá uma boa base a partir da qual iterar.

Os argumentos mais convincentes são do Marty Cagan (http://smashed.by/product-discovery), que diz:

> *Em vez de usar um prototipador por algumas semanas, a maioria das empresas usa o time de engenharia inteiro durante ciclos de releases completos para construir software que depois é testado e deployado nos sistemas de produção. É por isso que tipicamente as empresas levam de três ou mais releases durante um a dois anos para chegar a algo usável e útil. Eles estão usando a empresa de engenharia para construir um protótipo muito, muito caro, e usam seus clientes reais como cobaias involuntários.*

UCD é uma abordagem bastante conhecida que cria produtos mais bem-sucedidos por assegurar-se de que o design é feito para as necessidades dos usuários, e não por nossos desejos e caprichos. É uma abordagem iterativa que diminui os custos certificando-se de que as ideias são testadas com usuários antes de irem à produção.

UCD se baseia em alguns passos chaves, todos ligados às três classes de pesquisa que discutimos no capítulo *2. Descobrindo necessidades*, sobre necessidades de usuário:

- **Passo 1:** entenda necessidades. Comece com pesquisa exploratória para descobrir as necessidades não atendidas mais importantes dos usuários para resolver com o produto.
- **Passo 2:** crie conceitos e/ou protótipos. Crie rascunhos e protótipos para tornar as ideias tangíveis sem comprometer o precioso tempo de desenvolvimento.
- **Passo 3:** teste com usuários e itere. Rode testes de usabilidade nos conceitos e protótipos com cinco a dez usuários-alvo para validar as ideias e descobrir problemas de usabilidade (pesquisa de design). Volte ao passo 2 e repita quantas vezes for necessário ou possível.
- **Passo 4:** lance e meça. Desenvolva o produto, lance e meça o impacto que ele tem em métricas predefinidas (pesquisa de avaliação). Combine com métodos de pesquisa de design para descobrir por que as métricas subiram ou desceram. Faça mudanças conforme necessário.

Também é importante ressaltar uma das melhores coisas sobre UCD: ele se encolhe para se encaixar. Se você não tem muito tempo ou orçamento, você ainda pode fazer uma versão em menor escala para cada passo do processo.

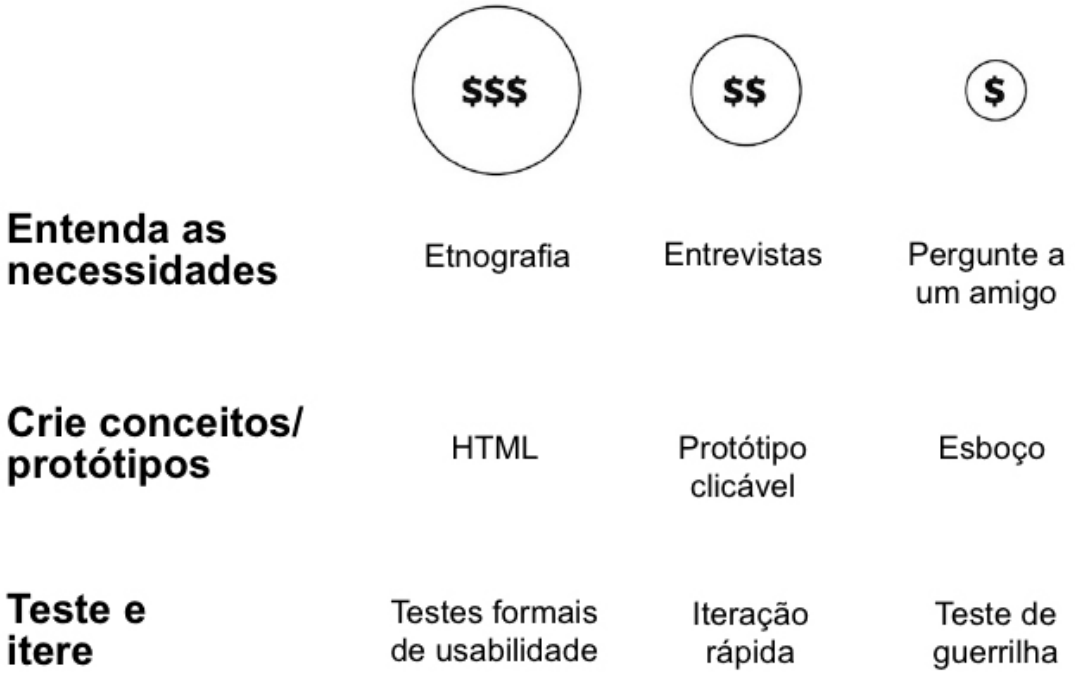


Figura 7.1: Design centrado no usuário se encolhe para encaixar em qualquer orçamento ou prazo

Recentemente, houve algumas críticas ao design centrado no usuário, e há argumentos de que precisamos começar a usar outras metodologias de design, tais como design centrado em atividade (*Activity-Centered Design*, ou ACD), junto ou no lugar de UCD. Cennydd Bowles resume bem os argumentos contra UCD em seu artigo *Looking beyond User-Centered Design* (http://smashed.by/beyond).

Os problemas incluem o fato de que ele leva mais tempo do que outras metodologias, nem sempre leva em consideração o estilo visual de um designer, e às vezes ele pode ser confundido como uma ciência que levaria à certeza (o que ele não é).

> Para uma boa visão geral do ACD, veja *Stop designing for users* de Mike Long, em http://smashed.by/stop-designing-users.

Como com a maioria das coisas, a metodologia precisa se ajustar à tarefa. Em alguns casos, UCD pode não ser o melhor encaixe, mas na maior parte dos projetos em que trabalhei é a melhor metodologia que temos para garantir que estamos projetando produtos úteis e usáveis.

Apesar de suas muitas vantagens, às vezes pode ser difícil introduzir UCD em uma empresa. Quando encontramos obstáculos, frequentemente culpamos a empresa por estarem obsoletas, ou por não entenderem por que é importante investir em bom design. Mas empurrar a culpa é uma abordagem errada. É dever nosso provar, dentro de nossa indústria e das empresas em que trabalhamos, que um bom design resultará em um retorno positivo sobre o investimento.

Se me lembro de uma coisa das primeiras aulas de Marketing, é essa óbvia, porém esquecida, verdade: os negócios existem para fazer lucro. Empresas trabalham duro para nos deixar feliz por apenas um motivo: para que compremos mais coisas delas.

Parece óbvio, mas é importante lembrar disso. Porque, quando você conversa com os times de gestão da sua empresa, contar pra eles quão bonito vai ficar o site quando você terminá-lo não será suficiente.

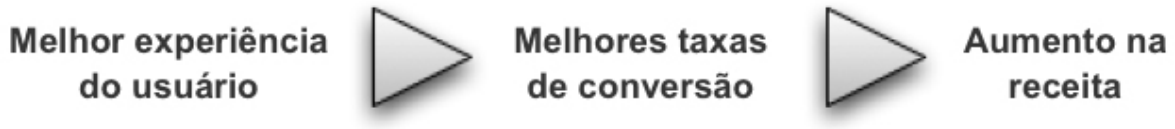
Para provar o valor do design centrado no usuário, você precisa provar que, investindo nele, o negócio trará mais dinheiro. É simples assim. Bem, o conceito é simples. A execução é... Complicada. Gostaria de propor algumas formas de nos ajudar a provar o valor do design, assim podemos passar mais do nosso tempo construindo ótimas experiências, e menos tempo contando às pessoas por que eles deveriam construir ótimas experiências.

Provar que um bom design fará mais dinheiro para sua empresa ou seu cliente não é algo que você consegue expressar em uma fórmula e da noite para o dia. Isso envolve um trabalho árduo e

pensamento inteligente (tempo bem gasto) para descobrir o que funciona mais para o contexto no qual você está projetando.

Em alguns casos, como projetar um processo de checkout, é relativamente fácil definir métricas e referências de sucesso e mostrar que uma reformulação resultou em mais dinheiro — porque é no checkout onde está o dinheiro.

Em outros casos, em que os fluxos estão mais afastados da geração de receita direta, pode ser muito mais difícil encontrar a ligação com o dinheiro. Nesses casos, um modelo de conversão costuma ser o caminho certo a seguir:



Na maioria dos fluxos de usuário, é fácil justificar que uma melhoria na taxa de conversão (ou redução na taxa de desistências) resultará em uma receita maior. Uma vez que esta ligação é feita, o que resta é provar que uma melhoria na experiência de usuário resulta em melhores taxas de conversão. Mas existem imediatamente dois desafios nesta abordagem:

- Como conseguir o tempo que você precisa para provar que UX funciona?
- Como encontrar o modelo correto de taxa de conversão para linkar UX às taxas de conversão (e, finalmente, receita)?

Vamos mergulhar nisso.

7.1 Conseguindo tempo para fazer a coisa certa

Fazer com que a empresa ou cliente gaste tempo com UCD pode ser um pouco complicado no começo. Sem dados reais, é difícil mostrar o valor dele. Mas sem o tempo para fazer o processo certo, é difícil conseguir dados reais. E então o círculo vicioso continua.

Uma forma de conseguir algum tempo é mostrar estudos de casos nos quais um investimento em UCD resultou em aumentos significativos na receita. Um desses estudos é o botão de 300 milhões de dólares de Jared Spool, em que uma simples mudança de design resultou em um enorme aumento na receita. Spool diz (*UIEtips: the $300 million button*, http://smashed.by/300-million):

> *Quando o time nos contatou, eles já tinham decidido qual era o problema e como iriam consertá-lo, mesmo sem nunca ter visto um cliente fazer compras. E eles estavam fatalmente errados. Não apenas a solução deles não ajudaria, mas nossa pesquisa mostrou que ela aumentaria a desistência.*
>
> *Duas semanas de testes de usabilidade ao vivo no site (e nos sites concorrentes), seguidos por duas semanas de testes iterativos de protótipos em papéis, produziram um processo simplificado de checkout, o qual, uma vez implementado, mostrou um aumento dramático nas receitas. É maravilhoso o que você vai aprender quando você realmente observar seus usuários.*

Em outro exemplo, a Airbnb mudou a maneira de adicionar propriedades em uma lista de desejo de uma estrela para um coração, o que resultou em um aumento de 30% em conversões (*How Airbnb evolved to focus on social rather than searches*, http://smashed.by/airbnb). E quando a Veeam Software mudou alguns textos de links de "Solicite um orçamento" para "Solicite o custo", seu ranqueamento de cliques aumentou em 161% (*How changing a single word increased click through rate by 161%*, http://smashed.by/click-through).

Obviamente, nem todas as mudanças em UX terão tanto impacto assim. Mas compartilhar histórias como esta, com a alta administração, vai ajudá-lo a defender o investimento em um devido processo de design centrado no usuário. Eu sei que é mais fácil falar do que fazer, mas montanhas podem ser movidas com alguns dados sólidos e uma persistência teimosa.

Então, vamos presumir que você provou que bom UX pode aumentar a receita significativamente. Como é que você vai provar isso para um de seus próprios projetos?

7.2 Os três As

Um princípio de marketing muito útil para UCD é olhar para a receita como algo vindo de uma ou mais de três fontes:

- **Aquisição:** fazer com que novos usuários se cadastrem em seu site ou serviço.
- **Ativação:** fazer com que esses usuários realizem suas primeiras compras.
- **Atividade:** fazer com que esses clientes de primeira viagem voltem para mais.

Se você puder atrelar um projeto de UCD a uma ou mais dessas três fontes de renda, mostrando que você aumentou as taxas de conversão nessas áreas, você conseguirá o que precisa. Você terá mostrado que design equivale a dinheiro. Eis alguns exemplos hipotéticos:

- *Um redesign do fluxo de registros* pode ser apresentado para melhorar a conversão entre a página de cadastro até usuários registrados. Isso está ligado à aquisição.

- Melhorias em uma *página de resultados de busca* podem ser apresentadas para melhorar conversões entre busca de itens até o carrinho de compras. Isso está ligado à ativação e à atividade.
- *Mudanças no layout da homepage e no conteúdo* podem ser apresentadas para melhorar as taxas de cliques em ofertas específicas a novos usuários. Isso está ligado à ativação.

E esta lista poderia continuar mais e mais. Não vai ser sempre fácil, mas todo projeto de UCD devia ser mensurável em termos do seu impacto no negócio. Ligar isso a um dos três As é uma boa maneira estruturada de ancorar todas as mudanças de design em objetivos de negócio, e finalmente, receita. Defina suas métricas de sucesso, marque-as como referência antes que alguma mudança aconteça, e então meça o (espero que melhorado) aumento nas métricas.

Designers precisam assumir um pouco da culpa por frequentemente terem de batalhar bastante pelos recursos apropriados para fazer nossos trabalhos. Precisamos entender por que os negócios existem, e seguir uma abordagem estratégica para fornecer o ROI de design.

- Apresente estudos de casos históricos para conseguir tempo e recursos para seguir um processo UCD apropriado em um ou mais de seus projetos.
- Comece com um projeto no qual mudanças podem ser medidas por uma melhoria em um dos três As da geração de renda: aquisição, ativação e atividade.
- Marque suas referências antes do começo do projeto, prossiga com o compromisso por UCD, e meça seus resultados. Veja este post no Six Revisions para algumas dicas sobre quais ferramentas de medição usar: *How to measure the effectiveness of web designs*, em http://smashed.by/effectiveness.

Este processo deve dar aos gerentes de produtos a oportunidade de introduzir o design centrado no usuário em uma empresa, de uma maneira mensurada e responsável.

Referências

BIAS, Randolph G.; MAYHEW, Deborah J. *Cost-justifying usability:* an update for an Internet Age. Morgan Kaufmann, 1994.

KARAT, C. Usability engineering in dollars and cents. *IEEE Software,* v. 10, n. 3, p. 88-89, maio 1993.

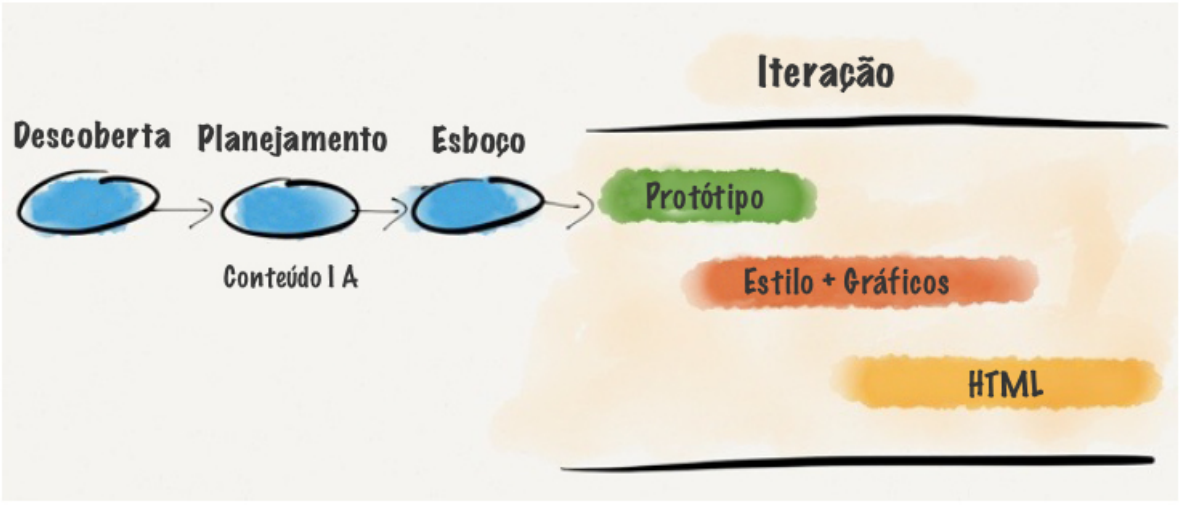
LEDERER, Albert L.; PRASAD, Jayesh. Nine management guidelines for better cost estimating. *Communications of the ACM*, v. 35, issue 2, p. 51-59, fev. 1992.

7.3 E quanto ao design responsivo?

Workflows: tornando UCD parte de todos os projetos

Uma das grandes questões em aberto na comunidade de design é como tornar o design centrado no usuário parte de nossos workflows, em um mundo cada vez mais multidispositivo. Este não é um livro sobre design responsivo — já existem muitos deles por aí. Mas acredito que é importante tratar sobre alguns possíveis workflows para os gerentes de produtos garantirem que projetos incorporem design centrado no usuário, enquanto também reconhecem o processo de desenvolvimento web.

Eis um workflow que funciona bem na agência em que trabalhei:



O objetivo desta abordagem é permanecer fundamentada em dois princípios centrais:

1. **Content first: primeiro, o conteúdo.** Precisamos parar de pensar no conteúdo em termos de layout, e planejá-lo independentemente do design.
2. **Mobile first: primeiro, dispositivos móveis.** Precisamos parar de focar no pensamento em dispositivos específicos, e admitir o mundo multidispositivo no qual trabalhamos nas direções de estilo, independente do layout.

Falarei brevemente sobre cada passo no diagrama e como eles nos ajudam a alcançar esses objetivos.

Durante a fase de descoberta, pesquisamos para descobrir necessidades do usuário, desenvolver personas e criar o mapa da jornada do usuário, que se torna a estratégia do nosso produto. Discutimos isso em detalhes na *Parte 2* deste livro.

Na fase de planejamento, evoluímos o mapa da jornada do usuário para um planejamento de conteúdo e um documento de informações de arquitetura. Discutimos isso na *Parte 2* também. Uma vez que temos nosso andaime no lugar, damos início ao processo de design.

Hoje em dia, raramente ainda fazemos wireframes estáticos, mas esboçamos bastante. Os benefícios dos esboços já foram comprovados várias vezes.

O que mais gosto no processo de esboçar é como ele permite que o time tente múltiplas soluções para um problema, antes de resolver fixar-se em uma ou duas ideias para iterar mais a fundo. Gosto de usar as planilhas de esboços responsivas de Zurb como templates, pois eles nos mantêm focados em uma abordagem multidispositiva (http://smashed.by/sketchsheets).

Uma vez que passamos pela fase de esboço com os clientes, e já sabemos qual abordagem gostaríamos de seguir, começamos a prototipagem. Usamos principalmente Axure, mas há muitas soluções por aí que se encaixam em uma variedade de abordagens.

Axure não é nativamente responsivo (ainda), então temos construído dois protótipos em nossos projetos: começamos *mobile first*; depois passamos para desktop. Isso não é o ideal, mas funciona para nossos propósitos atuais. Temos um forte foco em testes de usuário, portanto testamos esses protótipos em nosso laboratório de usabilidade, e iteramos o design com base nos descobrimentos.

Mais para fim do processo de prototipagem, começamos a trabalhar em *style tiles* (http://styletil.es), de modo que podemos conversar sobre a parte gráfica com os clientes, sem focar em questões de fluxo e layout. As style tiles ficam entre quadros de humor e amostras, nos quais são mostrados fontes, cores e elementos de interface, tais como botões e campos de formulários, mas não se preocupam tanto com layout.

Recentemente, esse método tem se expandido para incluir ideias como protótipos de estilo (uma renderização HTML responsiva de um style tile), e style tiles interativos. Veja mais em *Our new responsive design deliverable: the style prototype*

(http://smashed.by/style-prototype) e *Interactive style tiles — Short demo* (http://smashed.by/interactive-style-tiles).

Temos visto um enorme sucesso com essa abordagem. Uma vez que os clientes estão satisfeitos com a direção visual, o foco pode retornar à discussão sobre como a UI (interface do usuário) os ajudará a alcançar os objetivos do negócio e as necessidades do usuário. Isso também torna a transição de protótipo para design gráfico muito mais suave.

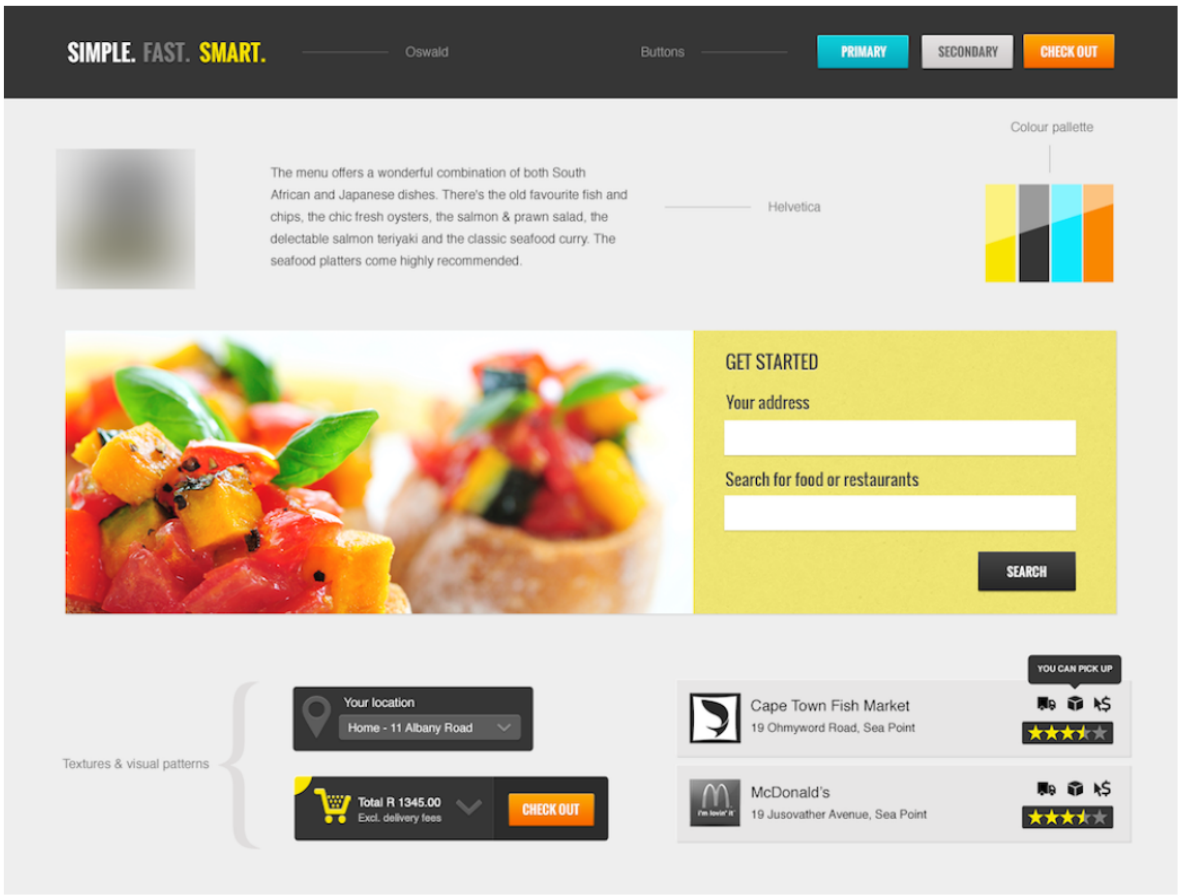

SIMPLE. FAST. SMART.
Oswald
Buttons
PRIMARY
SECONDARY
CHECK OUT
Colour pallette
The menu offers a wonderful combination of both South African and Japanese dishes. There's the old favourite fish and chips, the chic fresh oysters, the salmon & prawn salad, the delectable salmon teriyaki and the classic seafood curry. The seafood platters come highly recommended.
Helvetica
GET STARTED
Your address
Search for food or restaurants
SEARCH
Textures & visual patterns
Your location
Home - 11 Albany Road
Total R 1345.00
Excl. delivery fees
CHECK OUT
YOU CAN PICK UP
Cape Town Fish Market
19 Ohmyword Road, Sea Point
McDonald's
19 Jusovather Avenue, Sea Point

Figura 7.4: Um exemplo de style tile, mostrando a escolha de cores, botões, campos e fontes (por Alex Maughan — http://maughan.me/)

Existe certa preocupação de que os style tiles sejam uma enganação aos clientes, porque eles não mostram uma visão holística do canvas (veja *Designing in the transition to a multi-device*

world, de Francisco Inchauste, em http://smashed.by/multi-device), e é por isso que ainda não somos completamente pós-PSD (*The Post-PSD Era*, http://smashed.by/post-psd). Mas definitivamente nós não criamos o site inteiro no Photoshop.

Já que temos um protótipo interativo e fortes guias de estilo, geralmente apenas criamos por volta de seis páginas no Photoshop, de modo que os clientes possam sentir bem a direção (e o canvas holístico). E já que os clientes já estão tão familiarizados com o site, quando verem as compilações do Photoshop, geralmente por eles está tudo bem ver apenas algumas páginas, e fazer o resto do trabalho em código.

Na hora em que mergulhamos no Photoshop, nós geralmente já começamos a trabalhar no desenvolvimento front-end. Construímos os componentes do framework usando o protótipo e os style tiles, e ganhamos velocidade conforme o design gráfico é finalizado.

Não usamos frameworks clichês como Foundation e Bootstrap para código de produção. Neste ponto, concordamos com Aaron Gustafson, em *Responsive web design: 6 experts, 4 questions* (http://smashed.by/rwd-experts):

> *Eu acho que Foundation, Bootstrap e frameworks similares são interessantes do ponto de vista educacional, mas eu nunca os usaria para construir um site de produção. Para prototipar um conceito, claro, mas para levar um deles à produção, você precisa ser rigoroso na remoção de código CSS e JavaScript não utilizados, ou então você acabará criando uma experiência pesada e lenta para seus usuários.*

Um ponto importante sobre as últimas três fases: conforme o diagrama aponta, todas elas são fases muito iterativas. Realizamos mudanças a todo o momento, com base em feedbacks dos usuários, e conversas entre designers, desenvolvedores e o cliente. Acho que todos podemos concordar que design responsivo é

bagunçado, então precisamos estar confortáveis com uma certa quantidade de ambiguidade durante o design e o desenvolvimento. Está tudo bem, contanto que estejamos preparados para isso.

Tem sido um enorme processo de aprendizagem — e ainda estamos descobrindo as melhores formas de tornar o web design responsivo nossa abordagem padrão. Mas estamos comprometidos com isso, pois acreditamos em paridade de conteúdo (http://smashed.by/content-parity), e estamos convencidos de que design responsivo é a abordagem que nos levará até lá. Eis algumas coisas que aprendemos ao longo do caminho.

Primeiro, você não pode improvisar uma coreografia para o conteúdo. Não podemos simplesmente fazer nossos desenvolvedores front-end descobrirem o que acontece a cada breakpoint. Isso é algo que devemos planejar juntos para considerar todos os objetivos e limitações do projeto. Gráficos de breakpoints são particularmente úteis nesse passo (veja o livro de Stephen Hay, *Responsive design workflow*).

Segundo, otimize para *touch* e dê suporte a ações no teclado. Josh Clark aponta que "toda UI desktop deveria ser projetada para touch agora" (*New rule: every desktop design has to go finger-friendly*, http://smashed.by/touch-design). Ele está certo. As linhas entre o que é considerado desktop e o que é mobile estão borradas, então devemos presumir que tudo é *touchscreen*, e tornar os comandos fáceis de descobrir e manipular.

Os benefícios vão além do mobile. Colocar mobile em primeiro lugar nos ajuda a criar melhores sites desktop também, porque permanecemos focados em encontrar as necessidades de usuários centrais, e garantir que haja um caminho fácil e detectável entre os fluxos. Não há espaço para códigos estranhos e complexos em telas menores, e isso torna nossos designs para desktop melhores também.

Finalmente, é difícil, mas vale a pena. Como Ben Callahan aponta, em *The responsive dip* (http://smashed.by/responsive-dip): "O fato de que não sabemos como fazer algo hoje não significa que não devemos nos esforçar para fazê-lo amanhã." Ninguém tem tudo já determinado, então não sei quanto a você, mas eu quero ser parte da modelagem do futuro da web, não importa quão difícil seja.

Temos muito a aperfeiçoar, mas estou animado com o progresso que fizemos ao elevar o processo inteiro em direção à construção de sites responsivos. Cada projeto é executado de forma um pouco mais suave, e isso é encorajador. Portanto, meu único conselho àqueles que estão a beira do design responsivo é: mergulhe. Vale a pena.

Capítulo 8
Especificações

É hora de discutir a parte da gestão de produtos que gera medo no coração de qualquer pessoa que tenha trabalhado em uma grande empresa: o documento de especificações. Não importa como você o chama, a ideia é a mesma para todos os envolvidos: pilhas e pilhas de papelada desnecessária.

Mas não tema, vamos passar pelas más experiências e conversar sobre uma maneira de fazer a documentação, de forma que realmente nos ajude a criar melhores produtos, e não nos faça arrancar os cabelos.

Neste capítulo, discutirei três tipos de especificações:

- Especificações funcionais (como o produto deve funcionar).
- Especificações técnicas (como o produto deve ser implementado).
- Especificações de marketing (como o produto deve ser comunicado aos usuários).

Antes de continuar, é importante nos dirigirmos à pergunta: "Nós realmente precisamos de uma especificação?". Não posso falar melhor do que Joel Spolsky, então vou citar sua resposta por extenso (veja mais em *Painless functional specifications — Part 1: why bother?*, http://smashed.by/functional-spec):

Falhar ao escrever uma especificação é o único e maior risco desnecessário que você terá em um projeto de software. É tão estúpido quanto se dispor a atravessar o deserto de Mojave com apenas as roupas nas suas costas, esperando que dê para improvisar.

Programadores e engenheiros de software que mergulham na codificação sem escrever uma especificação tendem a pensar que são pistoleiros descolados, que atiram direto da bainha. Mas não são. Eles são terrivelmente improdutivos. Escrevem código ruim e produzem software de qualidade precária, e ameaçam seus projetos ao correr enormes riscos, que são completamente desnecessários.

Tendo dito isto, a maior parte dos documentos de especificação é ruim. São longos, chatos, e são feitos apenas para ticar um quadradinho dizendo que estão feitos. São escritos de uma vez e nunca são atualizados, e mais condenável ainda, eles não são usados durante o desenvolvimento.

Esta é uma situação que gerentes de produtos precisam desesperadamente evitar. Se uma especificação não está sendo usada ativamente durante o desenvolvimento, não é culpa do desenvolvedor: é culpa do gerente de produtos. É de responsabilidade do gerente de produtos entender que tipo de documento seria útil aos desenvolvedores, e então fornecer tal documento — algum que seja muito, muito melhor do que algo improvisado. É nisso que vamos focar neste capítulo.

1. **Especificações funcionais**

   A especificação funcional descreve como um produto funciona da perspectiva do usuário. O foco não é em como ele será implementado (isso é visto na especificação técnica), mas em definir fluxos e telas, e como será a experiência dos usuários com o produto.

Pode parecer um pouco acadêmico demais para alguns, e contra o espírito do movimento lean, que tanto defende se livrar dos entregáveis, mas temos de nos lembrar de que documentação não é ruim — documentação mal feita é ruim. Boas especificações funcionais ajudam times a se comunicar, a economizar tempo e a construir produtos melhores. Mas para garantir que suas especificações funcionais entrem na categoria de documentações boas, há alguns pontos importantes para lembrar.

2. **Especificações devem ser dinâmicas**

   Elas não são escritas de uma vez só e depois esquecidas. É por isso que especificações não deviam ser escritas no Microsoft Word (chega desses nomes de arquivo do tipo `v27_FINAL4.doc`). Em vez disso, usamos ferramentas colaborativas como uma wiki ou Google Docs para facilitar a edição e o acesso da versão mais recente.

3. **Especificações devem ser acessíveis**

   O documento de especificação não é algo que os gerentes de produtos escrevem isoladamente, antes de descerem a montanha e entregarem seus Dez Mandamentos para o time de desenvolvimento implementar. Qualquer pessoa na empresa deveria poder acessar as especificações a qualquer hora, e os membros do time deveriam poder fazer perguntas e contribuir com elas.

   Esse é outro motivo pelo qual usar o Word está fora de questão, e por que ferramentas online e colaborativas estão dentro. Sério, desinstale o Microsoft Word.

4. **Especificações devem ser flexíveis**

   O maior e mais válido criticismo com os documentos de especificação funcional é que eles são rígidos demais. A maioria deles é meramente uma lista de requerimentos que são

escritos por pessoas longe da implementação real, e uma vez que o trabalho está feito, são incapazes de adaptá-lo diante da realidade. Não é assim que eles deviam funcionar.

Sempre haverá alguma incerteza na especificação funcional, que só vai se esclarecer uma vez que o desenvolvimento começar. É uma coisa boa, já que o time inteiro está integrado com isso. Significa que times podem adaptar às necessidades dos produtos e dos usuários, e que eles estão dispostos a remover, alterar ou adicionar features se necessário (isto é, se a necessidade de negócio ou evidência de usabilidade está lá).

Então, o que vai em uma especificação? Esta é a única parte do livro onde as coisas podem parecer um pouco como um manual. Mas, por favor, continue com ela, porque ela é importante e vai tornar sua vida (e a de todo mundo que trabalha com você) muito mais fácil.

Eis os componentes básicos de uma boa especificação funcional. Para projetos menores, talvez você não precise de todos estes tópicos, mas mesmo assim é bom pensar neles, e usá-los como um template básico no começo de cada projeto.

Resumo do projeto

O resumo do projeto deve ser conciso e direto ao ponto, ao mesmo tempo em que comunica informações suficientes de modo que qualquer pessoa na empresa possa assumir o projeto, se fosse necessário, e soubesse exatamente com quem conversar e o que precisa ser feito. Tipicamente, as seguintes informações devem ser incluídas.

Contatos

Liste não apenas quem é o gerente de produtos no projeto, mas também o analista de negócios, líder desenvolvedor, líder de marketing, e por aí vai. Qualquer um que tiver uma voz no projeto deve estar listado lá.

Se há um patrocinador (um investidor ou de nível executivo), seus nomes também devem estar lá. Isso não apenas ajuda na comunicação, mas também garante que todo mundo que trabalha no projeto sinta o peso da propriedade e da responsabilidade necessário para fazer bons produtos.

Links

A maioria das empresas usa software de rastreamento para designar trabalho aos desenvolvedores. Podem ser desde ferramentas pesadas como Jira ou Pivotal Tracker, até uns mais leves como o Trello ou Basecamp. Às vezes, outras ferramentas de colaboração se misturam (tais como ConceptShare, para fornecer feedback sobre o design gráfico), então pode ser que fique realmente difícil manter tudo junto.

Felizmente, a maior parte desses programas possui um modelo de URLs que permite juntar todos os pontos de entradas à informação em um lugar central — a seção de links da especificação funcional. Liste lá todos os rótulos da implementação, designs técnicos, elementos visuais, conteúdo, comunicações de marketing, requerimentos de analytics e todo o resto. Isso permitirá a qualquer um mergulhar com facilidade em qualquer seção do projeto.

Definição do problema

Antes que qualquer detalhe seja mostrado, é importante descrever o problema que está sendo resolvido pelo projeto. Tudo retorna a isso. Se, depois de dois anos, aparece um novo investidor e pergunta por que o seu time fez algo (confie em mim, isso vai acontecer), a existência desta seção na especificação vai salvá-lo de muito retrabalho e dores de cabeça.

É espantoso o quão frequentemente as empresas revertem decisões, ou mudam a forma como fazem as coisas porque esquecem o motivo pelo qual o fizeram da primeira vez. Não deixe que isso aconteça com os seus projetos.

Objetivos de negócio

É essencial relacionar os objetivos do usuário, identificados na definição do problema, com os benefícios para o negócio. Como você espera que seja o impacto na receita, nas taxas de conversão e por aí vai?

Inclua os analytics e os dados sobre a situação atual e as alterações esperadas. É aí que os três As (aquisição, ativação e atividade) do capítulo *6. Design centrado no usuário e workflows* podem vir bem a calhar.

Métricas de sucesso

Como você vai saber que o projeto foi um sucesso? É importante determinar objetivos claros, mensuráveis e alcançáveis, com os quais todo mundo na gestão e nos times do projeto concordem. E não se esqueça de estabelecer marcas de referências antes que qualquer mudança aconteça, de modo que o progresso possa ser medido com precisão.

Análise competitiva

Se for o caso, forneça exemplos e análises de features ou produtos concorrentes que guiaram o projeto. É importante ter clareza nisso, não apenas para documentar ideias, mas também porque isso mantém o time honesto e focado em fornecer algum valor que seja distintivamente diferente de como os concorrentes fazem.

Escopo do projeto

Esta seção é usada não apenas para descrever o que é incluído no projeto, mas mais importante, para documentar o que está fora do escopo. Isso é importante porque os projetos frequentemente se tornam uma espiral fora de controle (chamada *feature creep*), a qual pode ser evitada com um claro entendimento de quais sistemas ou áreas do produto não são planejadas para serem alteradas no projeto.

O resumo do projeto é a metainformação sobre o projeto. O propósito é deixar todo mundo, do time e além dele, alinhado sobre os seus objetivos e limites. Também serve para construir uma memória organizacional, de modo que decisões não sejam revertidas no futuro sem um motivo ou planejamento devido.

Riscos e impacto em outros projetos

A próxima seção da especificação pega uma visão holística do projeto ao definir seu impacto em outros projetos e sistemas. Ela inclui, tipicamente, um esboço dos seguintes desafios em potencial:

- **Riscos conhecidos:** o que devemos reconhecer como potenciais efeitos em cadeia em outros projetos, tais como contratos que precisam ser assinados antes que qualquer coisa possa ser implementada, serviços que precisam ser construídos para dar suporte a nova features, e por aí vai.
- **Impacto administrativo:** o efeito do projeto em áreas fora de vista, que podem ser essenciais ao sucesso da execução. Isso inclui finanças, logística, distribuição, pagamentos e mais.
- **Impacto no suporte ao consumidor:** precisamos de uma nova documentação de ajuda? Agentes do serviço ao consumidor precisam ser treinados para tratar de alguma feature nova ou alterada? Isso é frequentemente uma reflexão tardia, e pode resultar em usuários irritados e em agentes de suporte ainda mais irritados.
- **Impacto na inteligência do negócio:** quais boletins adicionais são necessários para garantir que o novo produto ou feature possa ser rastreado de uma forma eficaz?

Uma vez que o resumo e os riscos do projeto foram delineados, é hora de entrar no processo de concepção e descrição da solução para o problema. Comece amplamente, e depois foque-se em partes específicas.

Jornadas do usuário e fluxogramas

Durante o processo de descoberta do produto, você deve ter criado mapas de jornada do usuário, e possivelmente algumas personas. Esse é um resumo visual do problema tanto quanto da solução. Logo, é uma boa ideia incluir esses artefatos na especificação, já que eles fornecem um contexto adicional sobre como o projeto se encaixa em uma visão mais ampla.

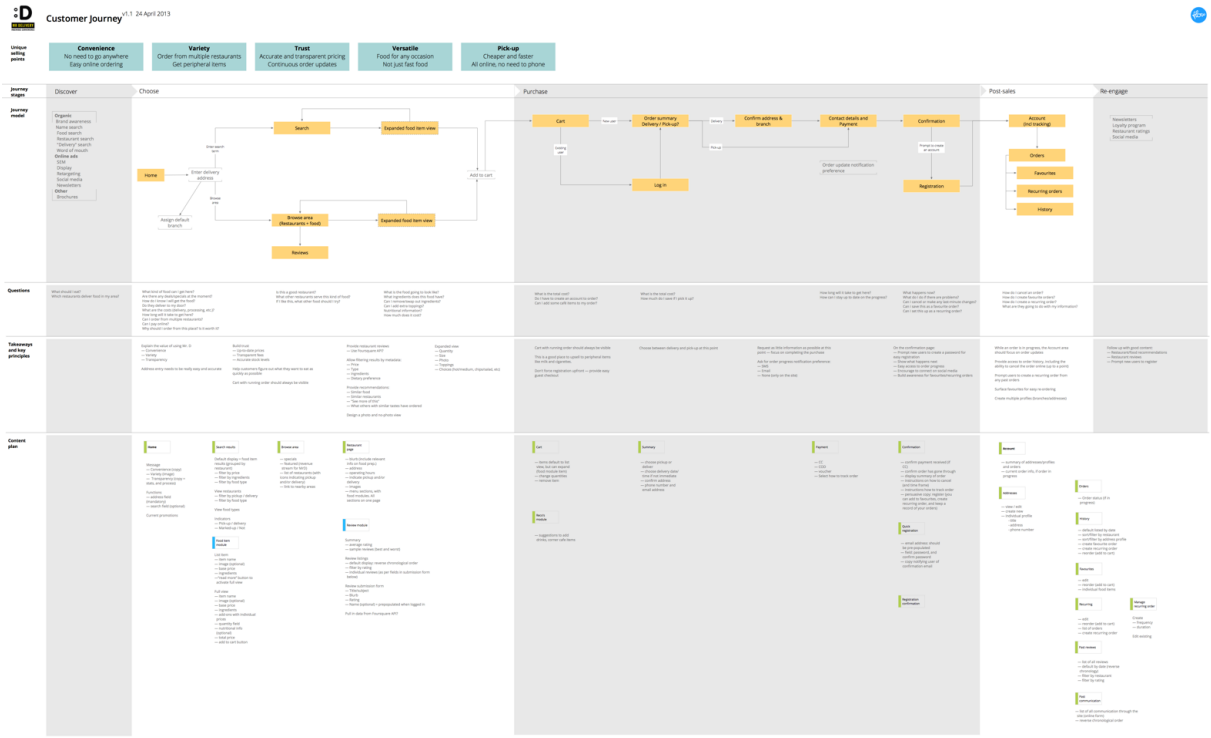


Figura 8.1: Mapa da jornada do usuário (veja em tamanho maior: http://smashed.by/customer-journey-map)

Este também é um bom lugar para incluir qualquer fluxograma que ajudará os desenvolvedores e testadores a completarem seus designs técnicos e planejamento de controle de qualidade. Isso é especialmente válido se há uma lógica complexa que dita os diferentes caminhos que os usuários podem tomar em um fluxo.

A maioria das pessoas deve estar familiarizada com os clássicos quadradinhos e setinhas nos fluxogramas, que parece com algo

assim:

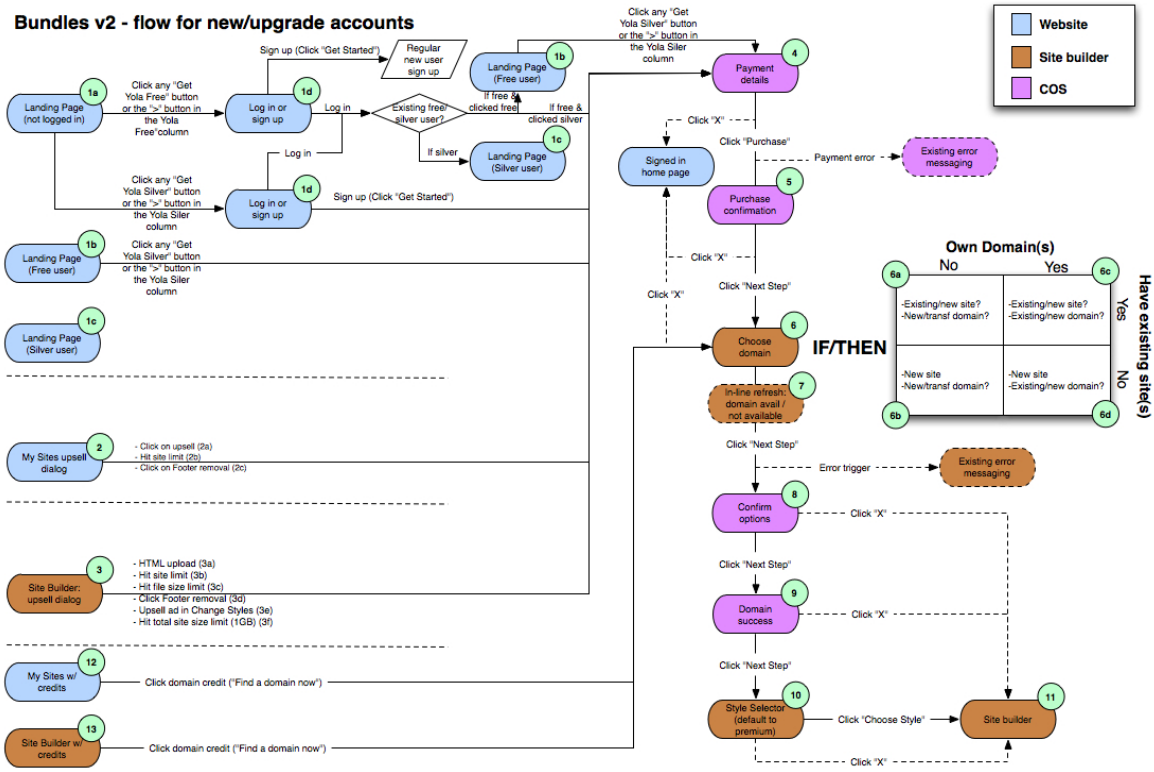
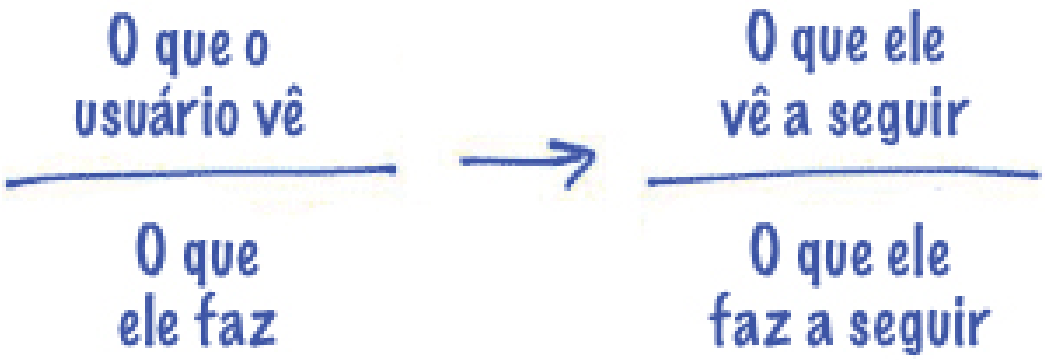


Figura 8.2: Exemplo de um fluxograma (veja em tamanho maior: http://smashed.by/flow-chart)

Esse formato definitivamente funciona bem, mas eu também comecei a usar o que Ryan Singer, do Basecamp, chama de fluxogramas *shorthand* (mais em *A shorthand for designing UI flows*, http://smashed.by/ui-flows), algo como uma abreviatura de um fluxograma:



Eis um exemplo de como ficará quando o completarmos:

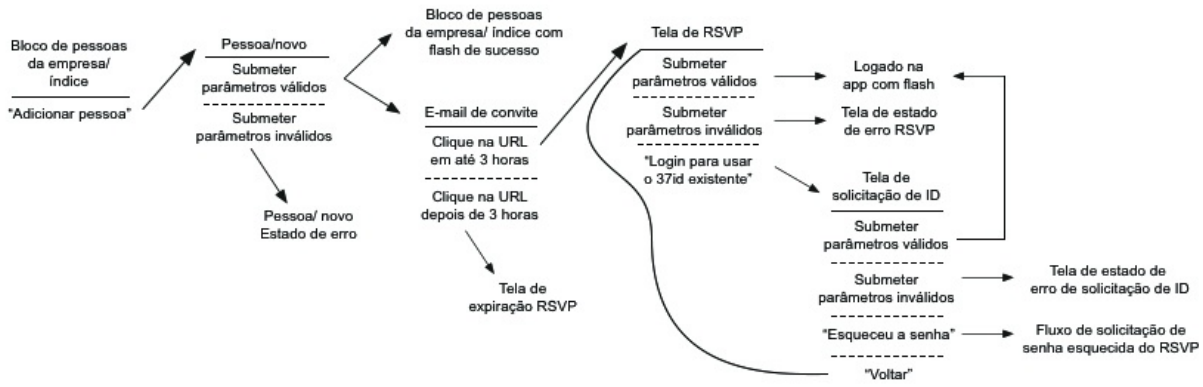

Bloco de pessoas da empresa/ índice
"Adicionar pessoa"
Pessoa/novo
Submeter parâmetros válidos
Submeter parâmetros inválidos
Pessoa/ novo Estado de erro
Bloco de pessoas da empresa/ índice com flash de sucesso
E-mail de convite
Clique na URL em até 3 horas
Clique na URL depois de 3 horas
Tela de expiração RSVP
Tela de RSVP
Submeter parâmetros válidos
Submeter parâmetros inválidos
"Login para usar o 37id existente"
Logado na app com flash
Tela de estado de erro RSVP
Tela de solicitação de ID
Submeter parâmetros válidos
Submeter parâmetros inválidos
"Esqueceu a senha"
"Voltar"
Tela de estado de erro de solicitação de ID
Fluxo de solicitação de senha esquecida do RSVP

Figura 8.4: Fonte para ambas as imagens: Ryan Singer, em http://smashed.by/ui-flows

Essa abordagem é particularmente útil porque trata bem especificamente dos chamados caminhos vermelhos (sendo o vermelho a cor do perigo, é claro) — aquelas partes de um fluxo onde usuários podem ficar travados ou cometer erros. Identificar os caminhos vermelhos em um fluxo o mais cedo possível ajuda por dois motivos.

Primeiro, força os designers a pensarem sobre projetar para prevenção de erros. Se você sabe onde um usuário pode fazer algo errado, é possível projetar um fluxo que os previna de cometerem esse erro.

Um exemplo simples é a prática de apenas mostrar um botão de `Próximo` ou `Continue`, uma vez que todos os campos obrigatórios em um formulário foram preenchidos. Assim evitamos a necessidade de apresentar mensagens de erros em submissão de formulários, já que ele não poderá ser submetido enquanto houver campos obrigatórios não preenchidos.

Segundo, os caminhos vermelhos nos forçam a pensar sobre manipulação de erros também. Eu sempre penso em manipulação de erros como o filho bastardo do UX design. Ninguém pensa nisso senão quando é tarde demais, e então tentam varrê-lo para debaixo do tapete. Mas boas manipulações de erros — como validações inline, mensagens que sejam claras e úteis — fornecem microinterações essenciais em uma boa interface.

Esboços, wireframes e protótipos

A indústria de UX está lentamente se afastando do uso de wireframes estáticos no design da interface do usuário e, na maior parte dos casos, isso é uma coisa boa. Wireframes estáticos são úteis para comunicar um conceito, mas eles frequentemente exigem muito esforço para que haja retorno, já que há modos mais rápidos de comunicar ideias (esboços), e melhores formas de testar a validade desses conceitos (protótipos interativos).

Portanto, na maioria dos casos, essa seção da especificação incluirá algumas fotos do processo de esboço para mostrar as ideias iniciais — especialmente quais diferentes tipos de ideias foram exploradas antes de escolhermos seguir tal direção. Variação *versus* iteração, lembra-se?

Aqui também geralmente incluímos um link para um protótipo interativo, construído em algum software como o Axure, ou frameworks como o Bootstrap ou o Foundation. O protótipo é frequentemente a parte da especificação mais especificada pelos desenvolvedores, já que é a mais útil.

Em vez de ter de ler documentos inteiros, ou perguntar a alguém toda vez que encontram um novo elemento na interface, os protótipos permitem que eles apenas cliquem e descubram qual é a intenção (e perguntem a alguém, se não estiver claro).

A especificação geralmente aponta para a versão final do protótipo, mas também é bom incluir links e referências a versões anteriores que foram usadas para validação do cliente. Como falamos no capítulo *6. Design centrado no usuário e workflows*, a fase de prototipagem é uma fase de hipótese, na qual diferentes ideias são testadas com usuários reais, e mudanças são feitas com base nesses feedbacks.

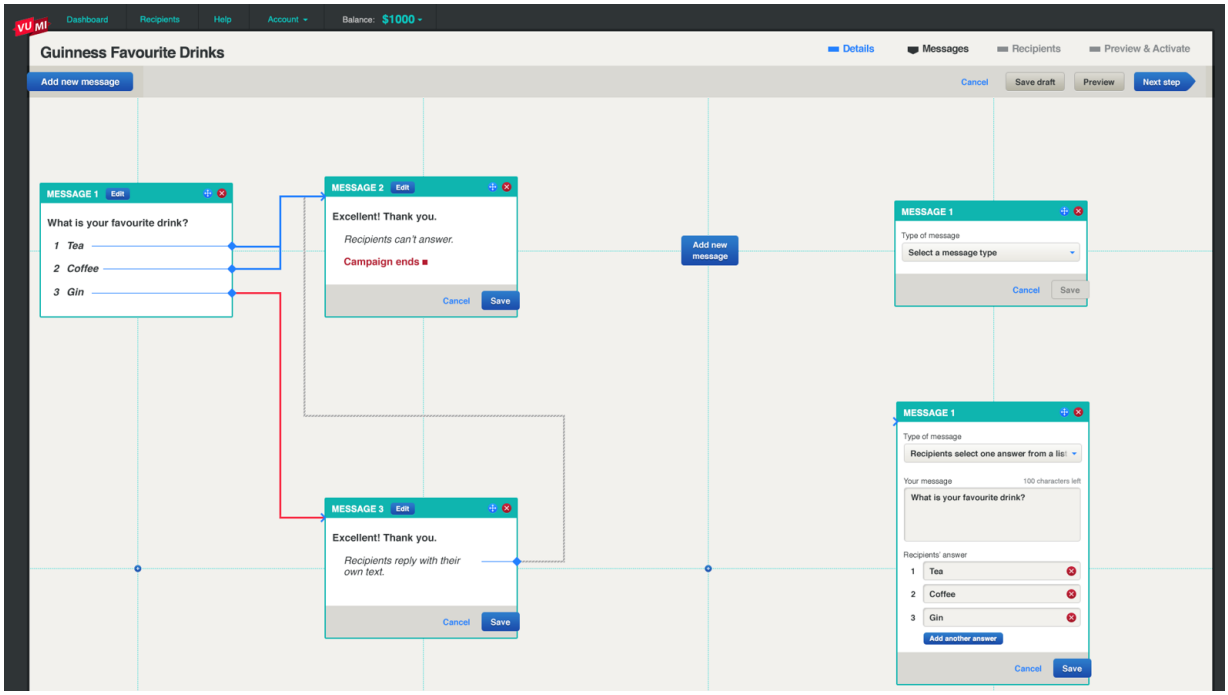


Figura 8.5: Exemplo de uma página de protótipo no Axure (veja em tamanho maior: http://smashed.by/axure-prototype)

Design gráfico

Essa parte da especificação comunica a direção visual do produto. Ela pode ter uma variedade de formatos. Como mencionado antes, não concordo completamente com a ideia de que estamos entrando

em uma era pós-Photoshop, mas acredito que estamos em uma era menos-Photoshop, em que não precisamos criar arquivos PSD para cada página ou tela do produto.

Em vez disso, uma combinação de um bom protótipo interativo, style tiles (http://styletil.es), e alguns arquivos-chaves PSD são mais do que o suficiente para os desenvolvedores codificarem as páginas. Style tiles são particularmente úteis porque separam a conversa sobre a aparência da interface ("Eu não gosto de amarelo!") da conversa sobre como ela funciona ("Esse botão está no lugar errado!").

Na minha experiência, se as discussões sobre o protótipo e a direção visual acontecem separadamente, é bem mais fácil fazer a transição do protótipo para as fases de design gráfico e de desenvolvimento, porque a maioria dos assuntos de aparência e sensação já terá sido resolvida.

O nível de design gráfico fornecido na especificação depende inteiramente da natureza do negócio, e de como os designers e desenvolvedores preferem trabalhar juntos (veremos mais sobre isso mais adiante). Para algumas empresas, arquivos PSDs comentados funcionam bem; outras preferem cada elemento separadamente e descrito em detalhe; alguns desenvolvedores estão contentes em mergulhar em arquivos PSD e criar uma biblioteca de componentes de UI com base no que encontram neles.

Não existe apenas uma maneira correta. Mas é importante que todo mundo tenha opinião sobre quais são essas entregas.

Bibliotecas de componentes estão além do escopo deste livro, mas são uma ótima forma de definir padrões visuais para um produto. Para um maior contexto sobre esta abordagem, veja o artigo *How and why to create a pattern library*, de Paul Boag, em http://smashed.by/pattern-library.

Os últimos vinte porcento

Com todas essas seções bem definidas, você terá uma especificação funcional que não apenas fornece o contexto do projeto e os objetivos (tanto do negócio como do usuário), que você está tentando resolver, mas que também é um documento realmente utilizado durante o desenvolvimento. Imagine! Mas há alguns apontamentos finais importantes de se fazer sobre especificações funcionais.

Primeiro, esse não é um documento que o gerente de produtos escreve no dia antes de o desenvolvimento começar. É um documento que tem início tão logo que um novo projeto começa.

Eu sempre crio um template em nossa wiki (ou onde quer que guardemos as especificações), e o abro logo que começo a trabalhar em um novo projeto. A melhor maneira de escrever uma especificação como esta é adicionar informação nela conforme estão disponíveis. Então adiciono o mapa da jornada do usuário assim que o completamos.

Também adiciono os esboços logo após a sessão de *design studio*. Isso reforça a ideia de que a especificação é um documento vivo, aberto à colaboração, e também reparte a carga de trabalho de modo que não seja um esforço enorme para criá-la.

Segundo, lembre de escolher apenas o necessário desta lista — nada mais. Para alguns projetos menores, eu pulo a jornada do usuário e a fase de prototipagem, e passo direto das mudanças no esboço para o desenvolvimento. Tudo bem fazer isso. Não pense nas seções anteriores como se fossem uma lei; pense nelas como um menu *à la carte*, no qual você pode escolher com base nas necessidades do projeto.

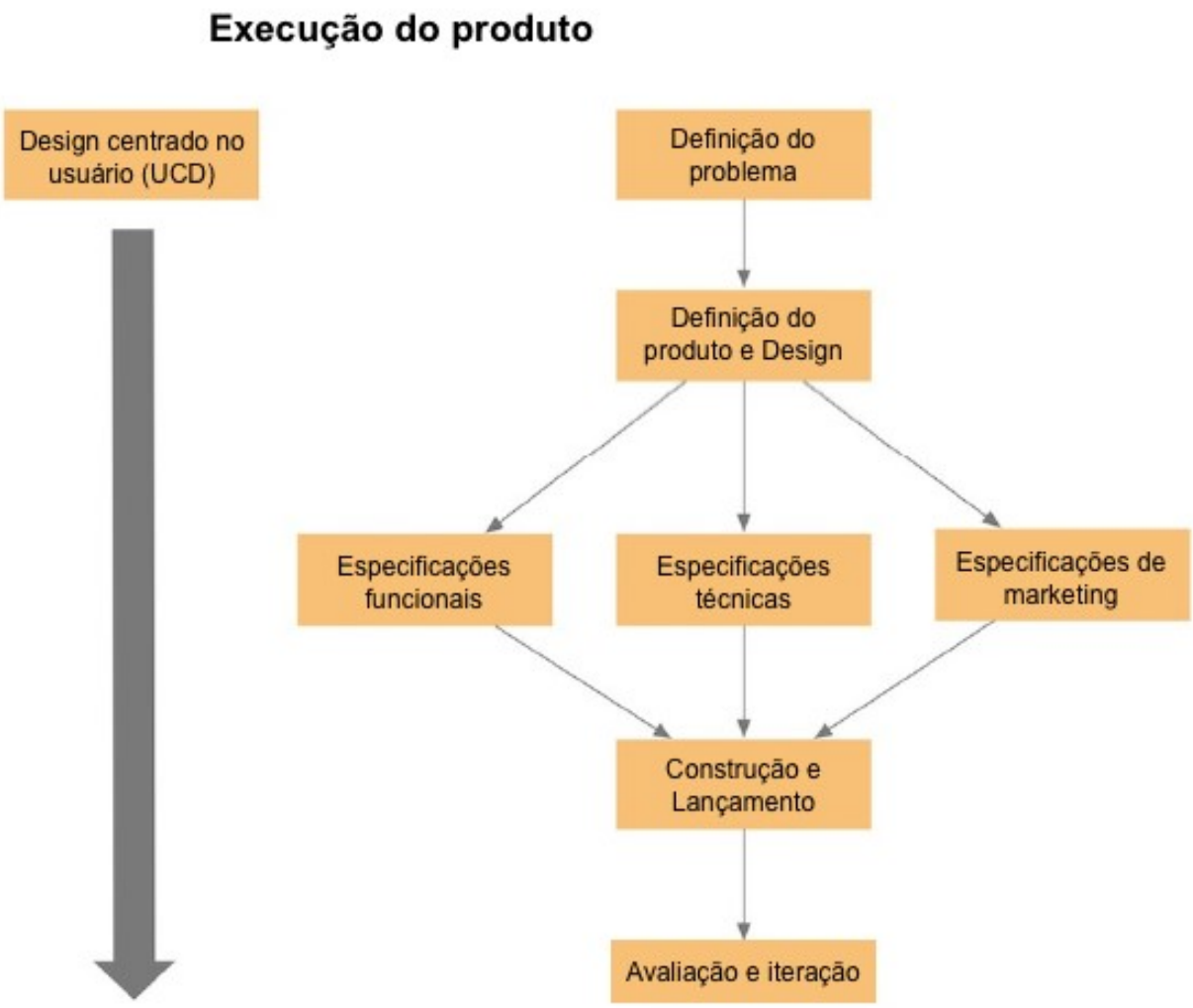
Finalmente, lembre-se de que especificações funcionais só vão lhe dar aproximadamente oitenta porcento do caminho até a entrega. Os últimos vinte porcento da definição do produto ocorre durante o desenvolvimento. É assim que deve ser.

A especificação fornece o contexto e a direção proposta, mas é só quando você começa a trilhar a estrada que você descobrirá as pedras no caminho, que precisam ser movidas ou evitadas. Abrace esta ideia, e esteja confortável ao negociar com designers e desenvolvedores para encontrar boas compensações.

Você quer evitar usar JavaScript em uma página, mas isso significa que você terá de posicionar absolutamente um elemento na página, o que traria consequências não intencionais no design? Bem, esta é uma compensação negociável entre melhoria progressiva e idealismo de design. O papel do gerente de produtos é entender essa balança, ouvir os argumentos de cada lado, e então tomar uma decisão e seguir em frente (e, lembre-se, conviver com as consequências).

Algumas pessoas viram a cara para as especificações funcionais, acreditando que ela faz parte da velha guarda da gestão de produtos, que não é mais relevante. Mas vou repetir o que eu disse antes: especificações não são ruins — más especificações são ruins.

Se você cria uma documentação que as pessoas realmente utilizam para construir o produto e entendem por que certas decisões foram feitas, como você pode argumentar que isso não é útil? Então, não deixe de escrever especificações. Apenas comece a escrever uma realmente boa.



Vamos tomar um tempo para nos lembrar de onde estamos no processo de execução do produto. Já discutimos a definição do problema e do produto, e depois de uma digressão sobre design centrado no usuário, conversamos sobre a importância de especificações funcionais. Agora vamos falar de outros tipos de especificações que também são importantes: técnica e de marketing.

8.1 Especificações técnicas

Enquanto a especificação funcional descreve como um produto funciona do ponto de vista do usuário, a especificação técnica descreve como o produto será implementado. Ela descreve coisas como estruturas de dados e modelos de banco de dados, e quais linguagens de programação e frameworks serão usados. A especificação técnica geralmente não é escrita pelo gerente de produtos, mas pelo líder técnico do projeto.

No geral, as especificações técnicas devem:

- Reiterar os objetivos do usuário e do negócio, como estabelecidas na especificação funcional.
- Definir a arquitetura e infraestrutura do sistema.
- Definir as tarefas de segundo plano necessárias para habilitar os fluxos de usuário determinados.
- Definir o modelo do banco de dados.
- Definir as interfaces com outros sistemas de back-office.
- Definir os requisitos não funcionais (liste coisas como o tamanho de página desejado, tempos de resposta, e por aí vai).

Gerentes de produtos que vêm de antecedentes não técnicos são frequentemente menos inclinados a se envolverem nesta parte do projeto. Embora eu não acredite que seja essencial para os gerentes de produtos saber como codificar, é importante que eles entendam o suficiente sobre arquitetura e infraestrutura tecnológica para ajudar a guiar as especificações técnicas.

Em alguns casos, programação adicional pode ser necessária para dar suporte a uma feature da forma como ela foi projetada. É, portanto, papel do gerente de produtos entender todas as negociações envolvidas.

O trabalho extra exigido resultou em uma melhoria significante na usabilidade, ou o mesmo objetivo pode ser alcançado por meio de

uma implementação um pouco mais fácil? O gerente de produtos pode não escrever a especificação técnica, mas eles a conhecem de cor e salteado.

8.2 Especificações de marketing

O envolvimento do time de marketing no desenvolvimento do produto é um assunto delicado. Frequentemente, existe um grande descompasso entre o que os designers do produto e os membros do time de marketing consideram importante.

O time de marketing se foca em fazer um produto que sejam tão bom que não será necessária nenhuma tática de vendas extravagante. A realidade é que essas equipes precisam achar um meio termo, porque precisam um do outro. Marketing não pode vender eficientemente um produto ruim, e designers de produto precisam aprender como comunicar os benefícios de seus produtos aos usuários.

Para assegurar que os times de marketing e de produto trabalhem bem juntos, é necessário que o time de marketing esteja envolvido em algumas fases apropriadas do processo de desenvolvimento. E o desenvolvimento dos requerimentos de marketing é o lugar perfeito para a colaboração. Os requerimentos de marketing especificam quem são os usuários de um produto em particular, e quais benefícios deveriam ser usados para vender o produto a eles.

A estrutura de que eu mais gosto para os requerimentos de marketing é a que a Amazon usa com frequência: escrever um curto comunicado de imprensa interno (não mais do que duas páginas) antes que qualquer design ou desenvolvimento comece. Esse comunicado permanece dentro da empresa, e não vai para nenhum veículo de mídia.

Em vez disso, ele é usado para ajudar os times internos de marketing a entender do que trata o produto, e como bônus também ajuda a manter os times do produto focados. Ian McAllister explica em *Ian McAllister's answer to What is Amazon's approach to product development and product management?* (http://smashed.by/mcallister-amzn):

> Comunicados de imprensa internos são centrados no problema do consumidor, em como as soluções atuais (internas ou externas) falham, e como o novo produto vai arrasar as soluções existentes.

Quando gerentes de produtos colaboram com o time de design em um comunicado de imprensa antes de o produto ser lançado, eles estão completamente alinhados quanto a para quem e para que serve o produto, e como ele será vendido aos usuários. Se qualquer coisa acontecer durante a fase de desenvolvimento para mudar um produto de modo que ele não esteja mais alinhado com o comunicado de imprensa, é uma boa hora de fazer uma pausa e descobrir o que precisa mudar: o comunicado de imprensa, ou a alteração planejada da feature?

Manter os requerimentos de marketing sincronizados com os requisitos funcionais e técnicos através do processo de desenvolvimento não só ajuda a construir relacionamentos saudáveis com o time de marketing, como também é outra verificação para garantir que você está construindo apenas produtos e features que realmente alcancem os objetivos de negócio e do usuário.

McAllister recomenda as seguintes seções no comunicado de imprensa interno, que se alinham muito bem com o processo que delineamos até agora:

- **Cabeçalho:** nomeie o produto de um modo que o leitor (por exemplo, seus usuários-alvo) possam entender.

- **Subcabeçalho:** descreva qual é o mercado para o produto e qual benefício ele terá. Apenas uma frase abaixo do título.
- **Resumo:** dê um resumo do produto e do benefício. Presuma que o leitor não vá ler mais nada, então este parágrafo deve ser bom.
- **Problema:** descreva o problema que seu produto resolve.
- **Solução:** descreva como seu produto elegantemente resolve o problema.
- **Uma citação sua:** uma citação de um porta-voz de sua empresa.
- **Como começar:** descreva como é fácil começar a usá-lo.
- **Uma citação do usuário:** forneça uma citação de um usuário hipotético que descreva como foi a experiência deles com o benefício.
- **Encerramento e chamada:** finalize e dê direções aonde o leitor deve ir a seguir.

Com esses três documentos flexíveis no lugar, é hora de começar a fase de desenvolvimento e a construir o produto.

Capítulo 9
Construção e lançamento

Quando é hora de começar o desenvolvimento, a maior parte dos blocos de construção para um ciclo de desenvolvimento bem-sucedido já deve estar no lugar. Você já terá determinado claramente as definições do problema e os objetivos, terá especificações funcionais, técnica e de marketing vivas, que definem claramente do que se trata o produto, enquanto ainda abre espaço para mudanças e improvisos.

Você terá um roadmap que não especifique prazos e datas de lançamento, mas mostre claramente a prioridade de quais problemas estão sendo trabalhados em qualquer ciclo dado. Então, encaminhar as coisas será fácil, certo?

Bem, não tão rápido. Todos nós sabemos como as coisas podem ficar loucas durante o desenvolvimento, e quão rapidamente começam a apontar os dedos quando algo dá errado. A verdade é que a maioria das pessoas tem medo de trabalhar com desenvolvedores, e isso afeta a qualidade do resultado, porque existem muitos desentendimentos no caminho.

Mas não é necessário ter medo dos desenvolvedores! Só é extremamente importante entender como desenvolvedores trabalham mais eficientemente. A boa nova é que, como Jeff Ello aponta em *The unspoken truth about managing geeks* (http://smashed.by/managing-geeks):

> *Diferente de muitas indústrias, a briga na maioria dos grupos de TI é sobre como fazer as coisas, e não sobre como evitar trabalho. Os profissionais de TI vão se auto-organizar, se desfazer e subverter em nome de concluir o trabalho.*

Uma vez que você entende isso, você também vai começar a perceber que desenvolvedores não odeiam processo. Eles odeiam processos que não ajudam a fazer as coisas. Então eles reagem bastante fortemente a coisas como reuniões inúteis com quinze pessoas e documentos de quarenta páginas no Word.

Para resolver este problema, as necessidades dos criadores (tais como designers e desenvolvedores) devem ser levados a sério por gerentes (aqueles que dirigem e disponibilizam o trabalho). É trabalho do gerente de produtos garantir que todo mundo na empresa entenda o trabalho dos criadores, e também definir as regras de engajamento com desenvolvedores para proteger seu tempo. Mike Monteiro entra nesse assunto atacando um humilde calendário, em *The chokehold of calendars* (http://smashed.by/calendars):

> *Reuniões podem ser tóxicas, mas a agenda é quem que permite que esse mal se prolifere. Todos calendários são uma porcaria. E todos eles são uma droga da mesma maneira. Agendas são um registro de interrupções. E muito frequentemente são um campo de batalha sobre quem é dono do tempo de quem.*

Paul Graham fornece uma visão mais holística em *Maker's schedule, manager's schedule* (http://smashed.by/makers-schedule). Ele explica que gerentes devem dividir seus dias em espaços de tempo de uma hora, enquanto criadores precisam de blocos de tempo mais largos para poder focar:

> *Quando você está operando no cronograma da criação, reuniões são um desastre. Uma única reunião pode eliminar uma tarde inteira, quebrando-a em duas partes, cada uma curta demais para se fazer qualquer coisa difícil. Além disso, você tem de se lembrar de ir à reunião. Isso não é um problema para alguém que está no cronograma da gestão. Há sempre algo a seguir na próxima hora; a única questão é o quê. Mas quando alguém no cronograma da criação tem uma reunião, eles têm de pensar sobre ela.*

Criadores precisam de longos períodos de tempo ininterruptos para fazer as coisas, e bem feitas. A maioria dos ambientes empresariais não dá suporte a isso devido a uma insaciável necessidade de que todo mundo concorde com tudo. Então ajudar as pessoas a entender por que isso é um problema tão grande para os criadores é importante, assim você pode efetuar mudanças culturais.

Michael Lopp fala sobre isso em seu artigo *Managing nerds* (http://smashed.by/managing-nerds). Substitua a palavra *nerds* neste artigo por "designers e desenvolvedores" (sem ofensas). Michael descreve como os nerds vão para sempre perseguir dois auges.

O primeiro é desfazer o nó: o momento em que eles descobrem como solucionar um problema específico ("Finalmente, uma forma simples de os usuários seguirem por este fluxo"). Mas o segundo é mais importante. É quando "a dominação completa dos nós" tem lugar — quando eles se distanciam dos dez nós desemaranhados, entendem o que criou os nós, e preparam-se para garantir que eles não aconteçam de novo ("OK, vamos construir um componente de UI que possa ser usado a qualquer momento em que essa situação ocorrer").

> *Perseguir o segundo auge é como os nerds conquistam seus salários. Se o primeiro auge é a alegria de entender, o segundo é o ato de criar. Se você quer que seu nerd sacuda seu mundo construindo algo revolucionário, você quer que eles persigam o segundo auge.*

E a forma de ajudar os designers e desenvolvedores a perseguirem o segundo auge é "proteger obsessivamente tanto o tempo quanto o espaço (deles)":

> *A missão quase constante do nerd é gerenciar toda a porcaria que nos está atrapalhando para chegar lá conforme procuramos pelos Auges.*

Então, como você pode mudar uma cultura erguida sobre reuniões e interrupções? Como você entende o que designers e desenvolvedores precisam para ser eficaz, e como você os protege incessantemente de distrações? Eis algumas maneiras de começar.

9.1 Pergunte aos criadores

Descubra o que designers e desenvolvedores precisam, e então faça acontecer. Um canto quieto no qual trabalhar? Claro. Uma tela maior? Absolutamente. Sem interrupções quando estiverem com fones de ouvido? Tudo bem. O que for necessário para ajudá-los a ser tão criativos quanto possível, e estarem livres para perseguir aquele segundo auge.

Também pergunte aos desenvolvedores como eles gostariam de estar incluídos no processo de desenvolvimento do produto. Alguns podem preferir ficar em seus cantinhos quietos e só escrever código. Mas também existem alguns que mergulhariam na

oportunidade de se envolver durante o processo todo — desde o conceito e design até o lançamento.

Quanto mais cedo os gerentes de produtos trouxerem os desenvolvedores para dentro do processo, melhor. Suas ideias e sugestões são inestimáveis, e eles também podem apontar possíveis restrições cedo o suficiente no projeto para economizar muita chateação e retrabalho ao longo do caminho.

9.2 Comece a trabalhar em uma cultura melhor de reuniões

Esta é uma constante batalha para empresas de todos os tamanhos, e há muitas formas de resolver isso. Tento aderir a duas simples regras.

Primeiro, uma reunião tem de produzir algo: um esboço, um plano de pesquisa, um design técnico, uma decisão estratégica para mudar o roadmap, e por aí vai. Segundo, sem mais grandes reuniões de status para atualizar o time de gestão sobre o que está acontecendo. É para isso que servem Google Docs e wikis.

Eu não iria tão longe quanto dizer que reuniões são tóxicas (veja *Meetings are toxic*, por 37signals, em http://smashed.by/meetings), mas quanto à necessidade de se ter uma agenda e um resultado tangível, isso nem preciso falar. Também gosto da proposta de Scott Berkun para lidar com reuniões recorrentes (*Do you need that meeting?*, http://smashed.by/need-that-meeting):

> *Mesmo lá na Microsoft, eu tinha esta regra quanto a reuniões recorrentes: no nascimento das reuniões, devia ser planejado que elas vão morrer. Vão deixar de ser úteis em algum momento. Mas muitos de nós sofrem por meio de reuniões zumbi, que perduram em um estado semimorto para sempre. Frequentemente, há uma pessoa que se sente poderosa na reunião e ele continuará a alimentar o zumbi com o cérebro de seus colegas de trabalho, apenas para preservar esse sentimento.*

Reuniões deveriam focar em facilitar coisas para as quais elas são boas: pensamento coletivo. Reuniões que me energizam são aquelas nas quais a maioria das pessoas está de pé, trabalhando junto em um objetivo comum.

Desde workshops de jornada de usuário até sessões de design studio para analisar os resultados dos testes de usabilidade, existem várias maneiras prestativas de usar nosso tempo em reuniões. Portanto, morte às reuniões zumbi. Vivam as reuniões que tornam os produtos melhores.

9.3 Ajude designers e desenvolvedores a se entenderem

Lucas Rocha fala da importância de designers e desenvolvedores trabalharem juntos e perto um do outro, em *Mind the gap* (http://smashed.by/mind-the-gap):

> *Processos de design iterativos que engajam designers e engenheiros desde cedo tendem a resultar em uma maior qualidade de UI, porque isso gera a flexibilidade e agilidade necessárias para misturar suas ideias conforme são implementadas. Parece óbvio, mas é muito mais fácil falar do que realmente fazer. Apenas veja quão raro é encontrar produtos com excelentes interfaces de usuário.*

Isso é muito verdadeiro, e o poder de times pequenos e colaborativos tem sido provado vezes e mais vezes. Mas é importante levar isso adiante. Não se trata apenas de colaboração, mas também trata-se de empatia. Se designers e desenvolvedores colaboram, mas não entendem uns aos outros, você ainda não vai chegar a lugar algum.

O principal problema é que designers e desenvolvedores abordam seus respectivos ofícios com perspectivas muito diferentes. Design é sobre composição — como colocar centenas de minúsculos elementos juntos de forma que o todo faça sentido. Desenvolvimento é sobre desconstrução — como quebrar o todo em partes que possam ser implementadas de forma eficaz. Isso gera uma desconexão que é difícil de ser superada se não houver empatia entre os dois grupos.

Thomas Petersen descreve a situação ideal muito bem em *Developers are from Mars, Designers from Venus* (http://smashed.by/mars-venus):

> *Os desenvolvedores saberiam o suficiente de design para apreciar o que um bom design pode fazer para seu produto, mesmo que isso signifique às vezes ter de desviar do framework e colocar um esforço extra para customizar certa funcionalidade.*
>
> *(...)*
>
> *E os designers aprenderiam a pensar como um programador quando fazem um design e desenvolvem uma estética que seja mais adequada para desconstrução em vez de composição.*

Então, não se trata apenas de fazer reuniões mais frequentemente. Trata-se também de se encontrar no meio do caminho para alcançar um objetivo comum juntos.

9.4 Celebre os sucessos

Dhanji R. Prasanna foi um desenvolvedor na Google Wave, e quando ele saiu da empresa escreveu um post excelente sobre alguns dos problemas em trabalhar em grandes empresas. Em *The mythical man-month* (http://smashed.by/mmm), ele diz:

> *Como um programador, você deve ter uma série de vitórias a cada dia. (...) É o que o deixa ávido pela próxima feature, e a próxima depois dela. E um time grande é um veneno para as pequenas vitórias.*
>
> *A natureza dos times grandes é tal que, mesmo quando você tem vitórias, elas vêm depois de longos, cansativos e muitos obstáculos desproporcionais. E isso tira todo o ar deles. Era comum, quando eu encaminhava uma feature, parecer mais um alívio do que euforia.*

É muito importante que times grandes celebrem sucessos com as pessoas com quem trabalham todos os dias. É difícil fazer isso certo em grandes empresas, porque a invisibilidade de membros individuais e as pressões para continuar para a próxima coisa não são naturalmente condutoras deste tipo de comportamento. Mas é possível se você trabalhar nisso.

Seja se você tem champagne em uma geladeira, se envia e-mails para toda a empresa agradecendo nominalmente as pessoas, ou toque um sino toda vez que um código é deployado (ok, este último é péssimo, desculpe), estar em uma grande empresa não é uma desculpa para agir como uma corporação impessoal.

Mais uma coisa: lembre de que o dia de release é algo importante. Como um gerente de produtos, você é o condutor em dias de release. Seja o primeiro no escritório (ou no chat) e o último a sair.

Garanta que todo mundo tenha o que precisa, e tome como único objetivo ajudar as coisas a correrem sem problemas. E não esqueça do champagne. O time merece uma vitória, então faça disto uma grande coisa quando o projeto toma vida — mesmo se você tiver uma release a cada duas semanas.

Uma das funções mais importantes de um gerente de produtos é apresentar progresso no produto, porque é o que motiva o time e mostra aos usuários que você está ouvindo e investiu nele. Então, não só planeje um monte de projetos. Entregue-os, tão frequente quanto possível.

Capítulo 10
Avalie e itere

Anteriormente no livro, falamos sobre a importância de definir métricas de sucesso antes que qualquer mudança ocorra em um produto — e garantir que haja referências a partir das quais você pode medir esse sucesso. Também falei sobre os três As, que podem ser bem úteis para ajudá-lo a identificar as métricas mais apropriadas: aquisição, ativação e atividade.

Uma vez que o projeto está vivo, é hora de fechar as contas. Tudo leva a este ponto para um gerente de produtos: descobrir se todo o trabalho duro prestado compensou, ou se você terá de fazer algumas mudanças para chegar aonde você precisa estar. Esse loop de feedback é importante porque não apenas determina sucesso, mas também ajuda a descobrir em quais áreas se focar a seguir.

Quando eu estava no ensino fundamental, uma das minhas estatísticas favoritas era (um pouco mórbido) relacionada a acidentes de carro. Quando ouvi que a maioria dos acidentes de carro acontece perto da casa das pessoas, não pude deixar de compartilhar esse fato aonde quer que eu fosse (*One in three road accidents happen a mile from home,* em http://smashed.by/accidents).

"É óbvio," — eu dizia — "as pessoas não prestam tanta atenção na direção quando estão quase chegando em casa". Foi só depois que eu percebi que o motivo pelo qual a maioria dos acidentes de carro acontece perto de casa é que a maior parte dos trajetos passam por perto da casa. Sempre penso nessa história como minha primeira exposição ao perigo dos dados, e quão facilmente mergulhamos em conclusões com um único ponto de dados. É por isso que é tão importante que o processo de avaliação inclua tanto medidas qualitativas quanto quantitativas.

A triangulação de pesquisa — também conhecida como pesquisa de métodos mistos — é uma metodologia muito útil para se ter em mente aqui. Triangulação de pesquisa é o processo de combinar vários métodos de pesquisa para estudar um único fenômeno (ou, neste caso, os efeitos de um único projeto).

Isso garante que cada método esteja em equilíbrio, de modo que, por exemplo, você não esteja dando muito peso nos dados de analytics e ignorando pesquisas ou feedbacks de testes de usabilidade. Geralmente, eu gosto de usar um equilíbrio de três métodos de pesquisas para determinar sucesso.

Web analytics

Meça as métricas pré e pós-lançamento com Google Analytics, ClickTale ou seu fornecedor de analytics preferido. São dados concretos, os números, a coisa que fundamentalmente conta para o sucesso do negócio. Entretanto, não é a história completa.

Pesquisas online

Quando eu trabalhava no eBay, rodei um programa chamado Product Tracker, que envolvia rodar uma série de pesquisas a cada trimestre, perguntando as mesmas questões, para medir como os sentimentos sobre cada um dos nossos fluxos mudava com o tempo. Desta forma, podíamos ver como os dados analytics correspondiam a atitudes e sentimentos sobre o produto.

Testes de usabilidade

Como eu continuo enfatizando, analytics (e pesquisas também) são ótimos para mostrar para você o que está acontecendo, mas você precisa de um método qualitativo como testes de usabilidade para entender por que algo está acontecendo. Essa é a única maneira de descobrir por que algo funciona (faça mais disto!) ou não funciona (faça diferente!).

Em um apelo passional para designers e desenvolvedores em um encontro em agosto de 2013, Stijn Debrouwere explicou como nossa obsessão por apenas analytics (sem outros dados) pode ser realmente prejudicial ao processo de desenvolvimento do produto. Falando especificamente sobre a indústria editorial, Stijn disse em *Cargo cult analytics* (http://smashed.by/cargo-cult):

> *Visualização de páginas é uma métrica vaidosa: algo que parece realmente importante, mas sobre a qual não podemos agir e não diz nada sobre quão bem realmente estamos indo, financeiramente ou o que quer que seja.*
>
> *(...)*
>
> *Não há nada como um painel cheio de dados, gráficos e linhas de tendência para nos fazer sentir adultos, como pessoas que sabem o que estão fazendo. Então, mesmo que não estejamos fazendo um uso real disso, é viciante e não conseguimos parar de fazer.*

Isso não quer dizer que analytics não são úteis. Mas temos de definir exatamente o que estamos medindo, e como isso nos ajudará a fazer produtos melhores. E depois, temos de triangular esses resultados com outros métodos para torná-lo realmente útil.

Existe uma grande cultura de testes A/B, ou teste e aprenda, na comunidade de desenvolvimento de software. Mas embora testes A/B possam ser extremamente valiosos para ajudar a identificar a melhor iteração para um site ou uma página específica, eles nunca devem ser usados isoladamente.

Já que testes A/B são relativamente baratos de se fazer e os resultados são tão convincentes, empresas passam pelo perigo de adotar uma cultura de *teste e aprenda*, na qual páginas são testadas apenas com A/B, sem entradas adicionais de usuários. Esse seria um jeito errado de fazer as coisas. Testes A/B não devem ser

usados por si só para tomar decisões, eles devem ser sempre usados em conjunto com outros métodos de pesquisa, tanto qualitativa quanto quantitativa.

Testes A/B são um método importante no conjunto de ferramentas de pesquisa, porque eles nos dão informações que testes de usabilidade sozinhos não conseguem. O principal objetivo dos testes A/B é ver como as métricas de negócio sobem ou descem, dependendo da versão da página — taxas de cliques, de checkout, de compras e por aí vai.

Isso você não pode ver apenas com teste de usabilidade sozinho. Mas, como Kohavi, Longbotham, Sommerfield e Henne apontam no artigo *Practical Guide to Controlled Experiments on the Web*, algumas das principais limitações dos testes A/B são:

- **Efeitos de curto versus longo prazo** — Já que os métodos de testes A/B geralmente usam experimentos controlados que ocorrem durante alguns dias ou semanas, alguns efeitos de longo prazo não são levados em conta, tais como usuários se familiarizarem com um novo design e, eventualmente, torná-lo o seu preferido. Esse é outro motivo para dar prosseguimento a pesquisas quantitativas e qualitativas adicionais por alguns meses depois que alguma alteração foi feita com base nos dados dos testes A/B.
- **Efeitos de primazia e de novidade** — Esses problemas representam dois lados da mesma moeda. Um efeito de primazia pode ser notado quando uma mudança em um padrão familiar faz que usuários experientes fiquem com incerteza e hesitação. Uma nova feature, por outro lado, pode encorajar alguns usuários a concentrar toda sua atenção nela. Tais questões sugerem que testes A/B devem ser rodados várias vezes durante um período mais longo, ou recrutar apenas novos usuários que não estarão sujeitos à influência desses efeitos.

- **Features devem ser implementadas** — Às vezes, a feature que está sendo testada não está completamente desenvolvida, já que é vista como um teste. Isso pode ser tendencioso para os dados: se a feature não tem uma boa performance, pode ser devido a uma execução pobre, e não porque a feature em si é ineficaz.
- **Consistência** — Devido à forma como a maioria das plataformas de testes A/B funciona, é possível que usuários vejam diferentes versões da mesma página em diferentes computadores. Isso pode gerar confusão para os usuários e, consequentemente, influenciar os dados.

O ponto de levantar essas preocupações não é dizer que não devemos usar testes A/B, mas que é importante usá-los com responsabilidade. Uma vez que cada método de pesquisa e de teste possui suas próprias limitações, uma combinação de métodos é a única maneira de se obter o cenário total e tomar as decisões corretas.

Um exemplo comum de como métricas de fonte única podem ser confusas é o tempo no site, uma métrica que a maioria dos sites segue religiosamente. O problema aqui é que é muito difícil saber se o objetivo deve ser aumentar ou diminuir o tempo que se passa no site.

Se os usuários passa mais tempo em um site, é porque eles estão mais engajados, ou porque não conseguem descobrir o que fazer? Se eles passam menos tempo em um site, é porque estão entediados, ou porque conseguiram alcançar o que queriam rapidamente? As métricas podem nos dizer que algo mudou, mas precisamos de métodos qualitativos para entender por que essas mudanças ocorreram.

É essencial levar o que você aprendeu de volta ao processo de desenvolvimento do produto. Portanto, tudo o que você aprendeu aqui retroalimenta a coleta de necessidades do usuário, do negócio

e técnicas que deram origem a tudo. E assim por diante, o ciclo continua: lance, aprenda, melhore. É disto que se trata ser um gerente de produtos.

Referências

KOHAVI, Ron; LONGBOTHAM, Roger; SOMMERFIELD, Dan; HENNE, Randal M. Practical guide to controlled experiments on the web: listen to your customers not to the HiPPO. *Data mining and knowledge discovery,* v. 18, issue 1, p. 140-181, fev. 2009.

Capítulo 11
Gestão de produtos em metodologias ágeis

Ao longo do livro, deliberadamente evitei falar sobre metodologias específicas de desenvolvimento, como agile ou cascata. Gestão de produtos é uma disciplina que se põe separada de qualquer metodologia em específico. São habilidades e métodos universais que podem ser aplicados independentes da filosofia que uma empresa utiliza para construir software.

Mas já que técnicas ágeis são atualmente a maneira mais comum que empresas escolhem para implementar seus projetos, gostaria de tocar brevemente neste assunto. Não vou passar muito tempo dissertando sobre a mecânica do desenvolvimento ágil — há outros textos muito melhores para isso. Esta definição de Anthony Colfelt é o suficiente para nossos propósitos (*Bringing User Centered Design to the Agile environment*, http://smashed.by/agile):

> *Agile é uma abordagem iterativa de desenvolvimento que dá pequenos passos em direção à definição de um produto ou serviço.*

Devido a essa abordagem iterativa, métodos ágeis se encaixam no modelo lean muito bem, e o processo possui vários benefícios importantes:

- O retorno sobre o investimento do desenvolvimento é mais rápido.
- Feedback constante para ajudar a melhorar o produto.
- Gera momentum para produto, o que é bom tanto para usuários quanto para times de desenvolvimento.

- Encoraja contribuições a partir de diferentes perspectivas, já que se baseia fortemente em times multidisciplinares (*cross-functional*).

Mas também há alguns desafios os quais superar, como Colfelt aponta:

- **Um papel não claro para o design.** Ainda não descobrimos como incorporar UX em ambientes ágeis. Existem algumas abordagens boas, mas ainda estamos tentando chegar lá.
- **O processo de descoberta do produto não é bem definido.** Técnicas ágeis focam em iteração, o que significa que raramente há tempo para definir um problema adequadamente, e tentar algumas soluções antes de escolher a iteração a qual seguir.
- **A tentação de dizer "está bom o suficiente".** A entrega supera a perfeição em agile, o que é ótimo, mas pode ser levado longe demais. A qualidade pode começar a cair para o acostamento completamente, no interesse em manter a velocidade, e isso é muito ruim para o produto.

O ponto importante para se lembrar é que metodologias ágeis são boas para refinar um produto, mas não para defini-lo. Invocando o linguajar que usamos ao longo do livro, agile é para iteração, não variação. Scott Sehlhorst também apontou um dos maiores perigos de agile em *The one idea of your product* (http://smashed.by/one-idea):

> *Agile não é um processo pelo qual você começa a digitar sem nem ter uma ideia daquilo que você pretende, lançar isso e então obter feedback em um processo iterativo. Se é assim que você está abordando o agile, seu processo está quebrado.*

Isso significa que o gerente de produtos preenche um papel crucial no ambiente ágil, primeiramente para assegurar que haja variação o suficiente no processo de design do produto, e não apenas iteração, e que haja um plano e uma visão claros. Em particular, eis algumas coisas as quais lembrar gerentes de produtos em um ambiente ágil.

- **O gerente de produtos é o Product Owner:** em times ágeis, o gerente de produtos preenche o papel do Product Owner. Isso pode parecer óbvio, mas algumas vezes os times trazem Product Owners para dentro, que não têm uma experiência em alguns dos outros aspectos que discutimos neste livro, como planejamento e estratégia. Isso é perigoso, porque...
- **O papel do Product Owner está entre as maiores responsabilidade do gerente de produtos:** um dos problemas que vejo na comunidade ágil é que os títulos de "gerente de produtos" e "Product Owner" são frequentemente usados indistintamente. Isso é um problema porque tem o potencial de confundir times, pensando que as únicas atividades pelas quais Product Owners são responsáveis são aquelas que fazem parte oficialmente da metodologia ágil, tais como a preparação do backlog.
- **Você ainda tem de planejar o futuro:** ser ágil não significa que você pode simplesmente seguir aos tropeções e esperar que tudo funcione. Não faz sentido encaminhar software funcional se for um software que ninguém quer usar. Isso não é falhar rápido, é correr um risco desnecessário. A visão e os roadmaps do produto ainda são importantes em um ambiente ágil.
- **Tente estar um ou dois sprints à frente do time:** há certa discordância sobre a ideia de sprint zero, e sempre garantir que o design e o produto estejam um pouco à frente do time. Mas descobri que se assegurar de que tudo o que for de UX esteja enfileirado — pelo menos oitenta porcento do caminho — antes que um sprint comece é a única maneira de garantir que os princípios essenciais de utilidade e usabilidade não sejam

descartados na intenção de entregar um produto o mais rápido possível.

- **Substitua especificações pesadas por especificações de alta fidelidade:** protótipos leves, documentados, como delineamos no capítulo *Especificações*, é a maneira correta de seguir com agile também.
- **Siga o espírito do agile, e não a carta:** é muito fácil ser pego pelas regras do agile. Já vi pessoas tomarem como uma religião a preparação do backlog, planejamento de sprint, retrospectivas e por aí vai. O espírito do agile é iterar e aprender seu jeito de fazer softwares melhores. O processo ajuda a levá-lo até lá, mas está tudo bem ignorar ou mudar algumas partes que não funcionem para o time.

A seguir, está um diagrama que mostra uma abordagem para incluir produto e trabalho de UX em um ambiente típico de scrum:

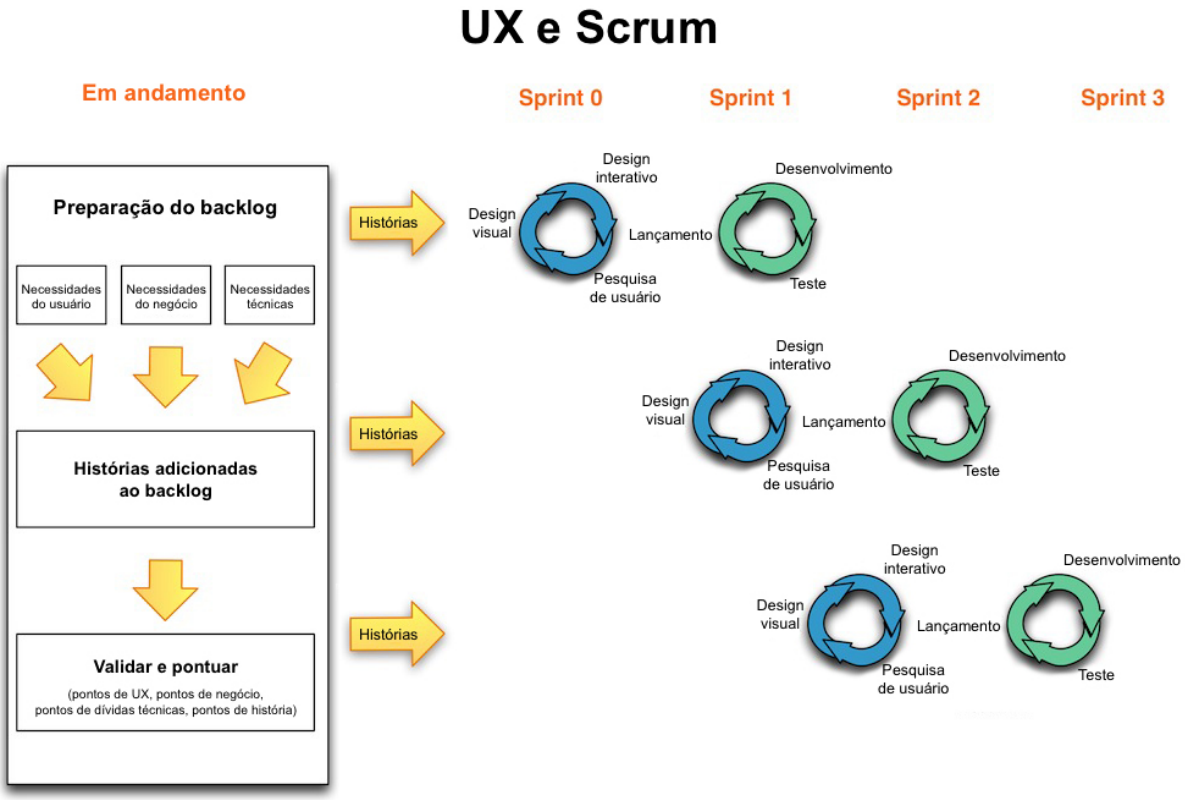


Figura 11.1: Uma abordagem para integrar design e princípios do scrum (veja em tamanho maior em http://smashed.by/ux-and-scrum)

Histórias são acrescentadas ao backlog com base nas necessidades do usuário, nas necessidades do negócio e necessidades técnicas que são descobertas durante o planejamento regular do produto. Até aí, tudo bem. A parte particularmente importante da perspectiva de um gerente de produtos é não apenas usar *story points* para pontuar cada história. As necessidades do usuário, do negócio e técnicas devem ter um impacto enorme na prioridade das histórias, então cada história deve também ser pontuada com base nesses aspectos.

A partir do backlog priorizado, eu ainda tive a experiência de que a maneira mais efetiva para manter o time de design envolvido ao longo do processo de entrega é garantir que testes de protótipo e de design aconteçam um ou dois sprints antes de cada fase de

entrega. Dessa maneira, o modelo genérico de gestão de produtos que compartilhei neste livro ainda pode ser aplicado efetivamente ao desenvolvimento ágil.

Capítulo 12
Começando

O último capítulo de um livro como este geralmente contém um resumo. Mas isso não parece certo, para um assunto que se trata tanto sobre fazer. Portanto, em vez disso, quero tratar de um assunto diferente para este capítulo, uma questão que ouço o tempo todo: por onde eu começo?

A teoria é boa, o framework é bom, mas como eu realmente mergulho nisso, e começo a ser um gerente de produtos? Como se percebe, isso também dá um bom resumo do livro, de qualquer forma, então esta situação será positiva para todos.

Chegar a uma empresa como um novo (ou, às vezes, o primeiro) gerente de produtos pode ser intimidador. Gestão de produtos geralmente é introduzida em uma empresa uma vez que existe um alto nível de entusiasmo interno e caos com o qual os líderes não estão mais sabendo como lidar. E então todos procuram o gerente de produtos para "gerenciar coisas".

É fácil se sentir sobrecarregado com tudo que há para fazer quando você entra em um papel estressante como gestão de produtos. Então, aqui estão algumas recomendações sobre como passar seus primeiros três meses em uma nova empresa.

12.1 Os primeiros 30 dias

Entenda o produto, o mercado e a cultura da empresa

Eis como eu defini gestão de produtos no capítulo *Papéis e responsabilidades do gerente de produtos*:

A missão do gerente de produtos é alcançar sucesso no negócio, atingindo as necessidades dos usuários por meio de contínuo planejamento e execução de soluções de produtos digitais.

Com isso em mente, passe os primeiros trinta dias aprendendo e entendendo:

- **O produto:** o que a empresa vende? O que o produto faz? Como ele funciona? Qual a proposta de valor? Quais problemas ele resolve para os usuários? Quais features ele tem? Que tipos de bugs ele tem? Quais são os principais problemas de usabilidade?
- **O mercado:** quem atualmente usa o produto? Como eles são? Quais suas características? O que eles gostam e desgostam no produto? Qual é o mercado-alvo? Há personas para cada tipo de pessoa diferente no mercado-alvo? Quais são as necessidades do macro e micromercados resolvidas pelo produto? Quem são os concorrentes?
- **O atual product/market fit:** você está em um mercado bom com um produto que pode satisfazer esse mercado? Quais são os buracos que você precisa tapar entre o que o produto faz e o que o mercado precisa, para garantir um melhor encaixe?
- **A cultura da empresa:** converse com o máximo de pessoas que puder na empresa — desde marketing, até finanças, design e engenheiros — para entender como as coisas funcionam. O que as pessoas gostam no processo de desenvolvimento do produto? O que elas odeiam? Os designers sentem que eles têm tempo suficiente para fazer seus trabalhos? Os desenvolvedores têm o que precisam para programar com eficácia?

Sobretudo, garanta que o papel do gerente de produtos seja propriamente entendido por todos. Como falamos anteriormente, para que um gerente de produtos seja efetivo, a empresa precisa entender que eles devem ter autonomia sobre os produtos que eles gerenciam.

O apoio de executivos é um pré-requisito para o sucesso, então assegure-se de que todos entendam que mesmo que todo mundo tenha uma opinião, nem todos podem tomar uma decisão. Lembre-se das palavras de Seth Godin: *"Nada acontece quando todo mundo tem de concordar".*

12.2 Os próximos 30 dias

Desenvolva um planejamento estratégico do produto

Com base no que você aprendeu nos primeiros trinta dias, comece a fase do planejamento do produto:

- Rode um workshop de descoberta do produto para começar a identificar as necessidades do usuário, necessidades do negócio e necessidades técnicas, bem como para criar um diagrama do quadro do problema.
- Desenvolva personas e jornadas do usuário, e comece a formular ideias para o desenvolvimento do produto junto do time.
- Trabalhe com o time para priorizar ideias e comece a construir um roadmap para o desenvolvimento. Considere métodos como o KJ-Method ou o modelo Kano, como maneiras de formalizar o trabalho de priorização.
- Identifique métricas de sucesso — defina como você saberá se o que você está fazendo está tendo o impacto desejado. Os

três As (aquisição, ativação e atividade) são sempre um bom começo.

- Coloque os devidos processos nos lugares certos para garantir que os ciclos de vida de desenvolvimento de produto sejam efetivos. Isso significa saber em que tipo de especificações os desenvolvedores precisam começar a trabalhar, como pesquisas e designs se encaixam no processo, como garantir que hipóteses e protótipos estejam sendo testados intrinsecamente na maneira como você trabalha, onde o marketing se envolve, como os testes de qualidade devem funcionar (o *quality assurance*) e por aí vai. Você só pode fazer isso uma vez que você entende a cultura corrente, e qual vai ser o planejamento estratégico.

Todos os itens anteriores vão dentro do planejamento estratégico do produto, como já definimos. Entre outras coisas, este planejamento inclui declarações sobre a proposta de valor do produto, quem é o mercado (perfis dos consumidores), como você planeja alcançar o encaixe entre produto e mercado (a oportunidade de negócio, precificação, distribuição), quais são as prioridades, um primeiro passo no roadmap, e as métricas de sucesso propostas.

12.3 Os últimos 30 dias

Comece a executar o planejamento estratégico do produto

Agora que o planejamento e o roadmap inicial estão prontos, comece a fase de execução do produto:

- **Comece com uma definição de problema razoavelmente pequena**, com métricas de sucesso claras e facilmente mensuráveis. Trabalhe com o time para fazer tudo certo

(trabalho em equipe colaborativo, aprendizagem constante através de testes de protótipo, especificações precisas).

- **Meça e mostre o sucesso do processo.** Use isso para construir confiança e continue a encaminhar melhorias (e produtos ainda melhores).
- **Avalie** a situação, e use o feedback dos usuários e do negócio para ajustar prioridades (e o roadmap) conforme necessário. Flexibilidade é a chave.
- **Siga em frente.** Repita qualquer um dos passos iniciais, conforme necessário.
- **Divirta-se** enquanto você estiver fazendo tudo isso.

Depois dos primeiros noventa dias, use o framework que falei ao longo do livro para pegar o ritmo de construir produtos ótimos: aprendizagem constante, testes constantes, sonhos constantes, criações constantes. Nunca haverá um momento de tédio.

O que eu apresentei neste livro é que a vida de um gerente de produtos tem um ritmo emocionante e exaustivo, que é tanto revigorante quanto é excruciante. Passar seu tempo movendo sistematicamente do planejamento do produto até a execução do produto, e mover-se entre essas fases continuamente, vão não apenas dar-lhe uma base sólida a partir da qual melhorar o produto, mas também garantirá que você encaminhe as melhorias certas no momento certo.

Existem milhares de maneiras de ganhar a vida. Mas nós, gerentes de produtos, escolhemos passar nosso tempo inventando produtos e levando-os mundo afora. Isso é um privilégio incrível, e uma oportunidade que eu não trocaria por nada.

Podemos trabalhar em compreender as pessoas, e descobrir como a tecnologia pode melhorar suas vidas. Sim, é estressante, mas vale

muito, muito a pena. Vamos lá fazer os melhores produtos que podemos fazer, juntos.